AMMRC Report No. CTR 76-21

COMPUTER-AIDED DESIGN AND MANUFACTURING FOR CLOSED DIE FORGING OF TRACK SHOES & LINKS

CARL F. BILLHARDT, NURI AKGERMAN, and TAYLAN ALTAN
BATTELLE, Columbus Laboratories
505 King Avenue
Columbus, Ohio 43201

July 1976

Final Report Contract Number DAAG-46-75-C-0041

Prepared for

ARMY MATERIALS AND MECHANICS RESEARCH CENTER
Watertown, Massachusetts 02172

AD

AMMRC Report No. CTR 76-21

COMPUTER-AIDED DESIGN AND MANUFACTURING FOR CLOSED DIE FORGING OF TRACK SHOES & LINKS

CARL F. BILLHARDT, NURI AKGERMAN, and TAYLAN ALTAN
BATTELLE, Columbus Laboratories
505 King Avenue
Columbus, Ohio 43201

July 1976

Final Report Contract Number DAAG-46-75-C-0041

Prepared for

ARMY MATERIALS AND MECHANICS RESEARCH CENTER
Watertown, Massachusetts 02172

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER AMMRC-CTR 76-21	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Computer-Aided Design & Manufacturing for Closed-Die Forging of Track Shoes and Links.		5. TYPE OF REPORT & PERIOD COVERED Final 11/11/75 - 5/10/76
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Carl F. Billhardt Nuri Akgerman Taylan Altan		8. CONTRACT OR GRANT NUMBER(s) DAAG46-75-C-0041
9. PERFORMING ORGANIZATION NAME AND ADDRESS Battelle's Columbus Laboratories 505 King Avenue Columbus, Ohio 43201		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS D/A Project: AMCMS Code: 3197.6D.4561 Agency Accession:
11. CONTROLLING OFFICE NAME AND ADDRESS Army Materials and Mechanics Research Center Watertown, Massachusetts 02172		12. REPORT DATE August 1976
		13. NUMBER OF PAGES 224
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer-Aided Design	Numerical Control
CAD/CAM	NC
Forging	Computer Graphics
Steel	Die Design

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

In this program, a computerized method has been developed (a) for designing blocker and finisher dies for forging track shoes and links, and (b) for manufacturing blocker dies via numerical control (NC) machining techniques. The computer-aided design and manufacturing (CAD/CAM) method, developed in this project has been incorporated in a system of computer programs called TRACKS. This system has been verified by using two military track-shoe forgings, the T-130 and the T-142.

TRACKS has the capability of designing dies for not only track shoes and links, but also for a myriad of nonsymmetric forgings, provided die design can be performed by analyzing the forging cross section by cross section. The results of the program indicate that CAD/CAM techniques can be successfully applied to designing of blocker and finisher dies for forging nonsymmetric components, which do not exhibit any modularity and do not belong to a geometric family.

FOREWORD

This final report on "Computer-Aided Design and Manufacturing for Closed-Die Forging of Track Shoes and Links" covers the work performed under Contract DAAG46-75-C -0041 by Battelle's Columbus Laboratories, from November 11, 1974, to May 10, 1976. To enhance readability, this final report has been written in the form of a summary text and detailed appendixes. The reader, interested in the details of the mathematical computations and computer program documentation, is invited to review the appendixes, which form a significant portion of this report.

The project was supported by the Army Materials & Mechanics Research Center (AMMRC), Watertown, Massachusetts, and by the U.S. Army Tank Automotive Command (TACOM), Warren, Michigan. The TACOM liaison engineer was Mr. G. B. Singh. The technical supervision of this work was under Mr. Roger Gagne of AMMRC.

This project has been conducted as part of the U.S. Army Manufacturing Methods and Technology Program, which has as its objective the timely establishment of manufacturing processes, techniques, or equipment to ensure the efficient production of current and future defense programs.

This program was conducted in the Metalworking Section of Battelle's Columbus Laboratories, with Mr. T. G. Byrer, Section Manager. The principal investigators of the program were Mr. C. F. Billhardt, Staff Scientist, and Dr. N. Akgerman, Principal Scientist. The program manager was Dr. Taylan Altan, Research Leader. Other Battelle staff members were consulted throughout the program as needed.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS
(Continued)

COMPUTER-AIDED DESIGN (CAD) AND COMPUTER-AIDED MANUFACTURING (CAM) FOR CLOSED-DIE FORGING OF TRACK SHOES AND LINKS

ABSTRACT

In this program, a computerized method has been developed (a) for designing blocker and finisher dies for forging track shoes and links, and (b) for manufacturing blocker dies via numerical control (NC) machining techniques. The computer-aided design and manufacturing (CAD/CAM) method, developed in this project has been incorporated in a system of computer programs called TRACKS. This system has been verified by using two military track-shoe forgings, the T-130 and the T-142.

TRACKS has the capability of designing dies for not only track shoes and links, but also for a myriad of nonsymmetric forgings, provided die design can be performed by analyzing the forging cross section by cross section. The results of the program indicate that CAD/CAM techniques can be successfully applied to designing of blocker and finisher dies for forging nonsymmetric components, which do not exhibit any modularity and do not belong to a geometric family.

INTRODUCTION

Many land and amphibious vehicles used by the Army require relatively large quantities of track shoes and links. These are made from various alloys of steel by closed-die forging, followed by finish machining. A considerable portion of the manufacturing cost of track shoes and links is incurred in producing the die sets (generally both roughing and finishing dies are needed), and in the material scrap resulting from the forging flash. Excessive die wear and unexpected breakage often add to these expenses.

The design of the dies, especially the roughing or blocker set, is an intuitive art, highly dependent on the skill and experience of the designer. After a best-judgment design is completed, a model is handmade in wood, plastic, plaster, etc. This is then traced by a tracer milling machine to produce the die cavity, or an EDM electrode from which the die is to be made. These various steps require skill and experience on the part of the craftsmen, and can take considerable time. After the forging dies are completed, additional time and money is needed to test and refine their geometry.

Among the factors to be considered, or calculated, in designing a forging process are the forging volume and weight, the volume distribution, flash dimensions, flash and scale losses, loads, stresses, and energies to be encountered, and the selection of forging equipment(1,2)*. These same factors must be considered for both preforming (often more than one) and finishing operations. Computations of variables, such as the volume or volume distribution are generally made by dividing the part into basic geometric subelements, such as spheres, cones, cylinders, rectangles, etc., for individual evaluation. Such procedures are inexact at best, require a great deal of time, and provide many opportunities for human error. Other variables may be determined strictly by judgment and/or experience.

The selection of equipment and the positioning of the dies under the selected forging equipment require the prediction of (a) the maximum forging load, (b) the forging energy, and (c) the center of loading in the die. The determination of center of loading and its positioning at the center of the press ram and bed reduces off-center loading and improves the tolerances in the forged part. These variables can be determined much more efficiently and accurately by using computerized procedures. Once the preforming and finishing dies are designed, computer techniques can also be used for manufacturing the dies by numerical control (NC) machining. Thus, die manufacturing cost are lowered and accuracy of the dies improved.

It has been shown that computer-aided design (CAD) and computer-aided manufacturing (CAM) techniques can be successfully applied to families of parts, such as turbine and compressor blades, to reduce the amount of judgment required to produce forging tools(3,4,5). Furthermore, these technologies can result in lower tool manufacturing costs, and better

---

* References are listed at the end of the report.

finished parts with less scrap losses. This project will extend the use of CAD/CAM to parts in the track shoe family.

The success of any manufacturing technology program depends mainly upon two factors:

(1) The quality and the usefulness of the work, from a technical point of view

(2) The acceptance, application, and use of program results by industry and others active in that field.

OBJECTIVES

The overall objectives of this program were: (a) to reduce manufacturing costs and improve productivity in forging track shoes and links, and (b) to reduce the lead times, which are traditionally long in supplying forged parts.

These objectives were addressed as follows:

(1) Reduced die manufacturing costs through the application of numerical control (NC) techniques and computer-aided design. At this time, dies are manufactured by copy milling a graphite electrode from a model and Electro-Discharge Machining (EDM) the dies. Copy milling takes longer time than comparable NC milling operations; consequently, application of Computer-Aided Design (CAD) and NC is expected to reduce die manufacture costs.

(2) Improved die life by more thorough analysis of the loads and stresses involved. Using the analysis of metal deformation in forging, the loading and stresses encountered during forging may be accurately predicted. Thus, it will be possible to design the die so as to obtain the maximum deformation at each state without exceeding the stress level which could damage the dies or the press.

(3) Improved dimensional tolerances and part-to-part consistency brought about by extended die life, and more exact duplication by NC machining when several die sets are required.

(4) Material savings by the reduction of flash losses. Through improved design of the dies and the forging process, less material will be lost into flash, as scrap.

With successful implementation of the results of this program, potential cost reductions (including material savings, die cost reduction, and reduction of machining) could be substantial since track shoes and links are produced in very large quantities. In addition, the potential benefits of the CAD/CAM techniques, developed in this program, include the improvement of dimensional consistency, and reduction of lead times for new and modified track-shoe designs.

PROGRAM HIGHLIGHTS

This program was conducted in 18 months. The following are the major tasks that were performed:

(1) Develop Computer-Aided Techniques for Die Design. Computer-Aided Design (CAD) programs were developed to automate the forging process design procedure. Given data as to geometry and material properties of the forging, these programs calculate such parameters as part volume, cross-sectional areas, perimeter, expected forging load, energy required as well as design of the preform. These programs were written in a generalized form. For designing the dies and calculating the process parameters, the user interacts with these programs via a graphics display terminal.

(2) <u>Develop Computer-Aided Techniques for Die Manufacture</u>. Once the preforming dies are designed via a computer program, since the geometry is already in the computer system, it is economically very attractive to develop the cutter paths for NC machining of the preforming die surface. Therefore, a system of computer programs were written that have the capability of automatically describing a surface, blending the various preform cross sections and calculating the cutter paths for NC machining of the surface.

(3) <u>Verification of the CAD/CAM Methods</u>. This task was carried out in parallel with the CAD and CAM system development described above. The T-130 track shoe was chosen as a typical part. Calculated loads were compared to loads measured in practice, computer-aided preform designs were compared to present designs and models were machined by Numerical Control (NC) machining.

(4) <u>Evaluation of Program Results and Generalization of Computer Programs</u>. This task was also performed in parallel with the others. Although the T-130 track shoe was used, the generality of the computer programs was always a major consideration. Towards the end of the project, the computer programs were applied to a T-142 track shoe in order to demonstrate the general applicability of the system of computer programs. Whenever possible, close cooperation was maintained with track-shoe manufacturers so that project efforts could be evaluated on a timely basis. Additionally, that familiarity resulting from this cooperation will lead to wider and faster acceptance of the methods developed.

GENERAL BACKGROUND ON COMPUTER-AIDED DESIGN AND MANUFACTURING (CAD/CAM) IN FORGING

Due to lack of well-established quantitative engineering methods, the computer has not been used extensively in forging applications. Recently, however, computers have begun to be increasingly applied to solve technical problems in forging process design and die manufacturing. Some of these applications are already routine operations in forging plants, others are used only at research and development laboratories or on an exploratory basis. Examples for uses of computers in forging are: (a) computer-aided cost estimating, where weight of stock, cost of die manufacturing, cost of auxiliary operations (such as shearing and heating of billets, trimming of flash, heat treating, cleaning, and inspection of forgings), and cost of forging operations are determined by a computer program, (b) use of computer programs in predicting load and energy requirements for a given operation, (c) computer-aided design of selected preform cross sections, (d) numerical drafting and NC machining of templates, and (e) NC machining of forging dies or graphite electrodes for EDM of the dies.

Recently, the Air Force Materials Laboratory sponsored a Manufacturing Technology Program on the "Application of CAD/CAM Techniques in Forging of Aircraft Structural Parts"(6). The main purpose of this program was to develop a computer-aided method for (a) designing the forging process, particularly for the preforming and the finishing dies, and (b) NC manufacturing of the forging dies. The vast difference between the geometries of aircraft structural parts and track shoes precluded the use of portions of the work reported in Reference (6). However, this prior work was very useful as a guide in formulating the overall conceptual approach to realizing a successfully operating CAD/CAM system. Contrary to aircraft structural parts, track shoes do not exhibit geometric modularity; therefore, their analysis and preform design requires a more global approach.

Computer-Aided Design of Preforming Dies

In order to achieve adequate metal distribution, many closed-die forging operations need intermediate steps, commonly known as blocking or preforming. In the final forging operation, defect-free metal flow, complete die filling and minimum flash loss can be achieved only by proper design of the preforms.

In many forge shops, preform or blocker dies are designed by experienced die designers mostly by intuition and by using empirical guidelines. Recently, preform design has been computerized for rib-web type forgings(6).

A detailed review of many rib-web type structural parts reveal that almost all structural cross sections can be divided into basic components of L-shapes, as shown in Figure 1, where 10 different L-shapes form the cross section of a forging. Hence, once a generalized design procedure for the basic L-shape is set up, the computer-aided technique can easily combine the basic modules in a building-block manner to obtain preforms for many parts. Further, by modifying certain design parameters, such as the ratio of the web thickness of the preform to that of the finish part, different preform shapes can be designed for different materials to be forged in different forging machines.

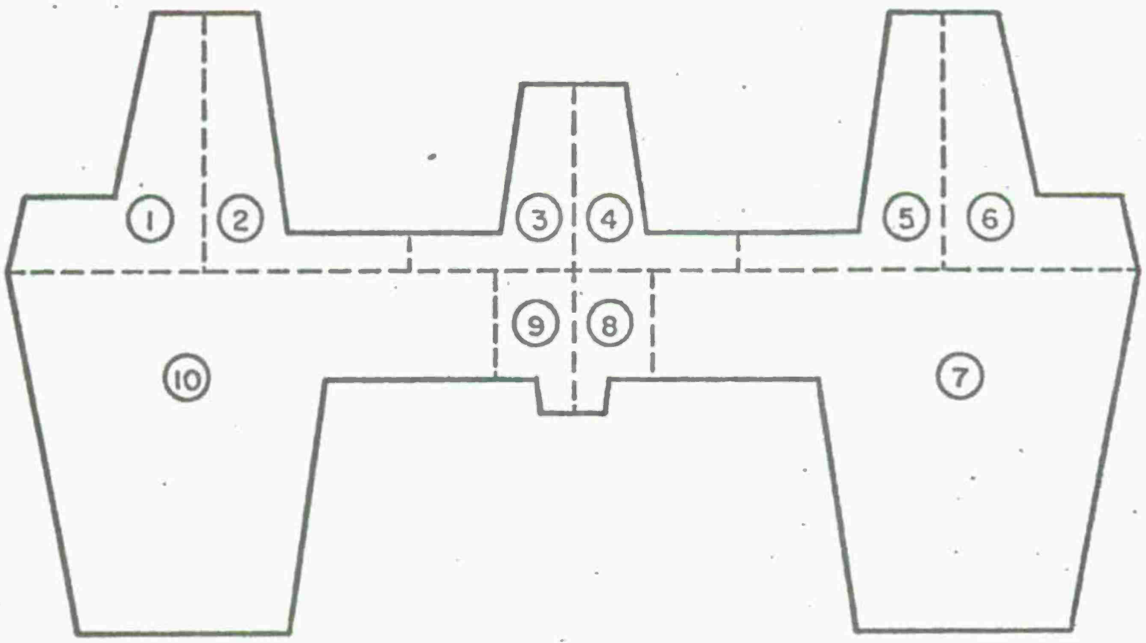


FIGURE 1. DECOMPOSITION OF A RIB-WEB TYPE FORGING CROSS SECTION INTO ITS COMPONENT L-SHAPES

Since track-shoe cross sections do not exhibit the modularity of most structural parts, one is forced to assume a global approach to preform design and consider a given cross section as a whole. A literature survey was conducted comparing finish die designs to the corresponding preform designs in order to develop mathematical guidelines suitable for automating preform design. These guidelines yield an acceptable preform most of the time, although not always (see Appendix VII). Therefore, extensive interaction capability was implemented so that the experience of the user can be brought to bear on the computations to obtain an acceptable preform. The computational speed of the computer and the visual perception of the designer complement each other in producing a preform design.

Computer-Aided Design of Finish Forging Dies

Apart from the geometry of the forging, it is the flash configuration that determines the finish die geometry. Flash dimensions directly influence the stresses and loads encountered by the dies during forging. When the stresses exceed the allowable levels, dies either crack or deform permanently. By varying the width and thickness of the flash, the stress distributions and the peak stress can be varied. A thicker flash will result in a lower peak stress. A shorter flash land will also have the same effect.

In determining the flash dimensions, the designer makes a compromise between the maximum allowable stresses on the dies, the capacity of available equipment, and the amount of excess flash material needed to fill the die. This decision is best made by using an interactive graphics terminal which displays the stress distribution and the average pressure, as seen in Figure 2 for a track-shoe cross section.

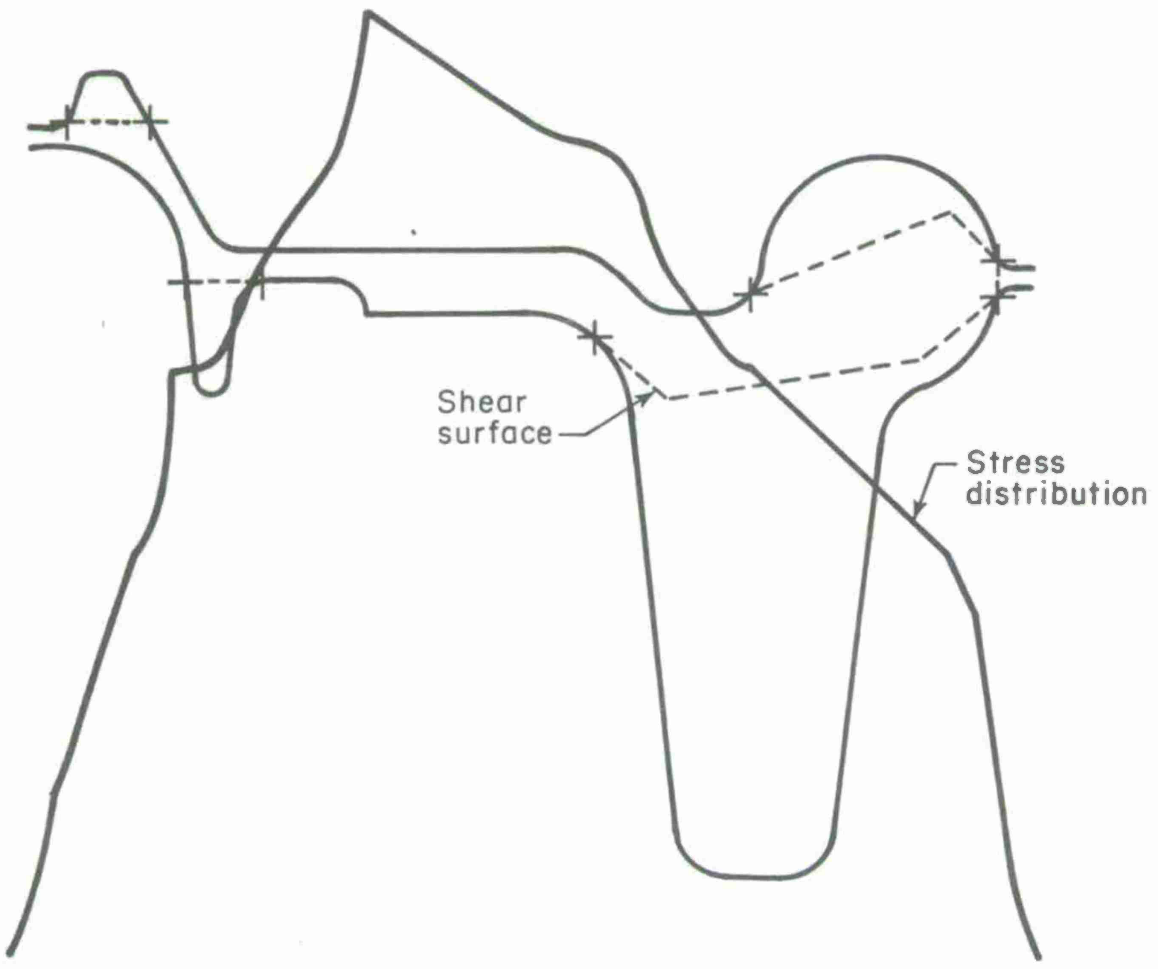


FIGURE 2. TYPICAL FORGING CROSS SECTION, THE SHEAR SURFACES OF THE FLOW MODEL (BROKEN LINES), AND THE CALCULATED STRESS DISTRIBUTION FOR A GIVEN MATERIAL AND FLASH DIMENSIONS

Manufacturing Forging Dies by Numerical-Control (NC) Machining

The forging dies, or the electrodes for EDM of these dies, are usually manufactured by copy milling. For this purpose, it is necessary that a skilled model maker first makes a model from wood, plaster, or plastic. At the present state of the art, NC machining is most economical for parts of the same geometrical family and for certain parts exhibiting symmetry. The economics of NC are apparent especially when NC and conventional die-sinking techniques can be combined. In this case, the majority of the die surface is machined by NC while the details of machining the corner or the fillet radii can be carried out conventionally. The most significant characteristics of copy and NC milling can be compared as follows:

Characteristic/Application	Copy Milling	NC Milling
Parts with symmetry	Special machine or model necessary	Same NC tape/simple program change
Changes in part	Difficult to introduce	Easy to program
Delivery for dies	Long	Short
Cutting time	Long	Short
Accuracy/Reproducibility	Acceptable	Excellent
Shrinkage	Included in Model	Simple to program
Set-up time	Long	Short
Space requirements	More	Less
Skill is with	Model maker	NC programmer

When preform design is performed via computer programs, it is economically very attractive to go one step further and produce a tape for NC machining of the preform surface. This economic advantage is based on two observations: (1) The geometry is already in computer readable form, and (2) the geometry is defined by cross sections which are to be "blended" by the model maker. Thus, a generalized blending algorithm can take care of most of the surface.

Computer programs were written which process the output of the preform design stage defining blends between cross sections, and calculate the cutter paths for NC machining of the EDM electrode for the preform die. Frequently, track shoes and links exhibit at least one axis of symmetry, thus enhancing the economics of NC.

DEVELOPMENT OF THE CAD/CAM SYSTEM FOR TRACK-SHOE FORGING

Program Approach

The CAD/CAM approach used in this program is outlined in Figure 3. The system of computer programs developed is called TRACKS. The use of TRACKS for designing forging dies starts with the description of the forging. The data base consists of cross sections of the forging. These cross sections in turn are expressed in terms of x,y,z coordinates and the associated fillet or corner radii. These data are first operated on by the preprocessor, which translates them into internal canonical forms and calculates such geometric parameters as cross-sectional area, perimeter, etc. These preprocessed data are optionally operated on by the analysis section or the preform design section, or both. The operation and use of the system of computer programs TRACKS are detailed in Appendix I.

Given the material properties under the contemplated forging conditions, the analysis section can calculate the expected stress distribution for various flash configurations. The maximum stress is typed out and the stress distribution is displayed graphically. The predicted forging load, the energy required and the average pressure are also calculated. Maximum stress, average pressure or load can be used as a criteria for choosing a suitable flash geometry. The method used in calculating stresses and loads is detailed in Appendix III. Energy calculations are based on the technique explained in Appendix V.

The preform design section, although capable of designing a preform automatically in some cases, has extensive interactive capabilities for modifying a given geometry. The experience and visual perception powers of a die designer is coupled via interactive graphics to the computational power

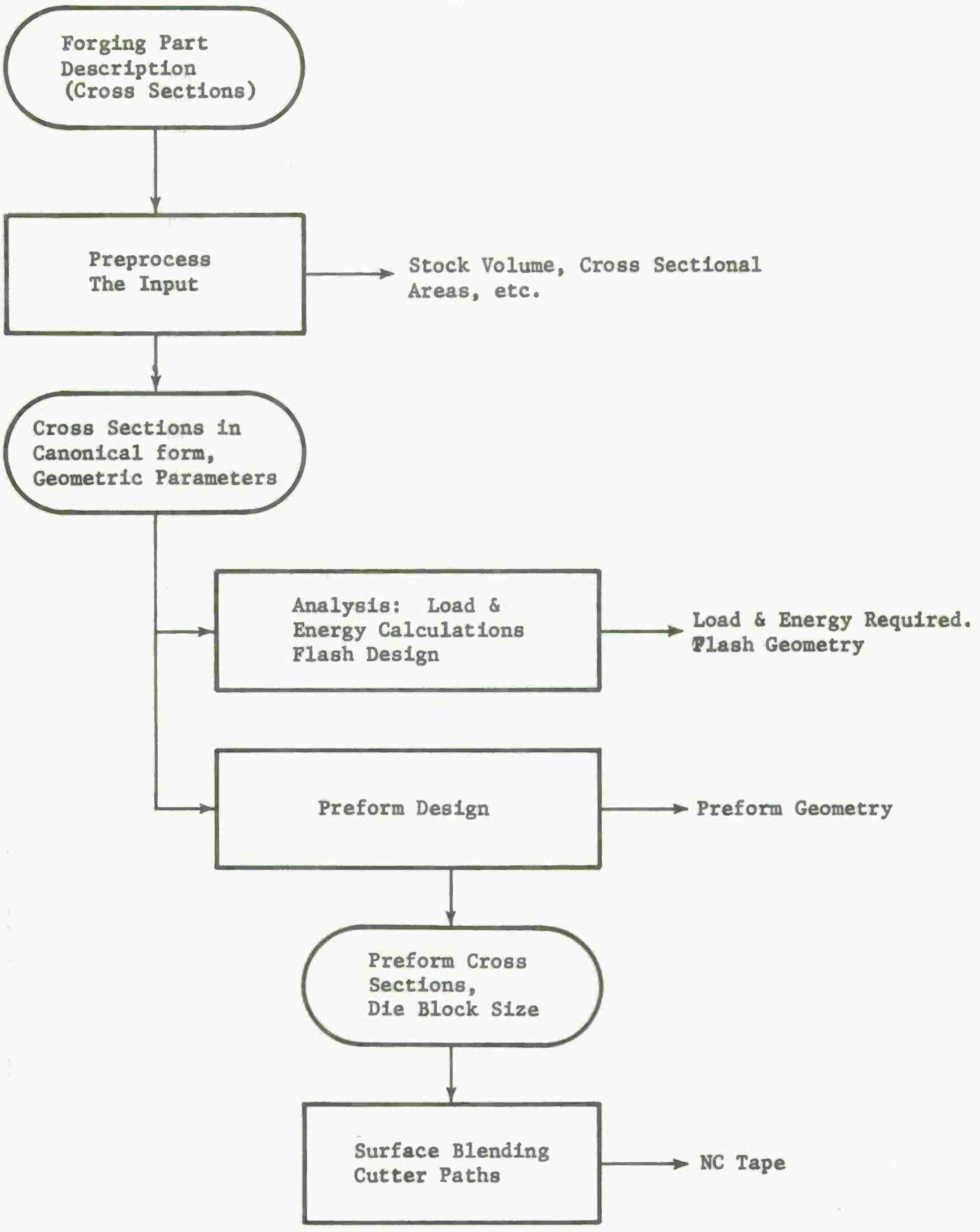

Forging Part Description (Cross Sections)
Preprocess The Input
Stock Volume, Cross Sectional Areas, etc.
Cross Sections in Canonical form, Geometric Parameters
Analysis: Load & Energy Calculations Flash Design
Load & Energy Required. Flash Geometry
Preform Design
Preform Geometry
Preform Cross Sections, Die Block Size
Surface Blending Cutter Paths
NC Tape

FIGURE 3. FUNCTIONAL OUTLINE OF THE CAD/CAM SYSTEM FOR TRACK-SHOE FORGINGS

of the computer. Through interactive graphics, the preform design is modified until the designer is satisfied with the result.

The output of the preform design section is saved for subsequent input to the NC section where preform cross sections are blended together and cutter paths are calculated. A complete NC tape file is generated for machining the EDM electrodes to sink the preform die cavities. The techniques used in blending the cross sections are explained in Appendix IV.

System Concepts

Early in the program, it became apparent that perception played a vital role in preform design. Die designers were basing their decisions not only on volume distribution, but also on the relationship between the geometric features of a given cross section. Perception is nearly impossible to program, especially when the geometry under consideration does not exhibit any modularity as in track-shoe cross sections. Designers, on the other hand, can perceive the subtle geometric relationships between the features of a cross section at a glance. Thus, interaction between the man and the machine is required. This can best be achieved via interactive computer graphics. The best hardware for this task is a refreshed graphics tube with a light pen. Such an interactive graphics terminal might be connected to a time-sharing service. However, due to the high volume of data required for graphics, in order to be responsive, such a system should possess high data transmission speeds. High data rates are usually not readily available on time-sharing systems. Instead of a large time-sharing computer, a minicomputer could be used. This choice, although slower in performing arithmetic, has the following advantages:

- Stand alone unit, dedicated to shop operations
- Extremely powerful graphics capability
- Relatively inexpensive hardware.

Thus, the CAD/CAM system for track-shoe forgings was implemented on a minicomputer system. The major hardware components, shown in Figure 4, necessary for successful operation are:

FIGURE 4. PDP-11/40 MINICOMPUTER SYSTEM WITH REFRESH GRAPHICS DISPLAY TERMINAL USED IN DEVELOPING THE "TRACKS" SYSTEM OF COMPUTER PROGRAMS

- PDP-11 with at least 28k of memory
- One disk drive
- VT11 display processor and graphics tube
- Keyboard terminal.

Hardware and operating system software requirements are detailed in Appendix VI.

System Capabilities

The actual software employed reflects the functional outline of Figure 3. As is the custom in conventional die design, the CAD/CAM system, called TRACKS, also works with cross sections. Initially, a data file containing the sections that are to be considered during design, is created.

Given a forging drawing, a die designer selects the sections he wants to work with. A draftsman would then obtain the coordinates describing these cross sections. Another means of obtaining accurate descriptions of the cross sections would be an APT part program. (APT is the major part programming language for NC machining). This approach would be attractive especially if the finish die is to be NC machined. Then parts of the same APT program can be used for both NC machining and sectioning purposes.

Design is performed one cross section at a time; sections need not be ordered consecutively on the data file. Once a cross section is read in and processed, it is displayed to the user, who then has the option of stress analysis, preform design, switching to another section or stopping. Similarly, in each phase, various options are presented to the user. It is possible to cycle through the various phases testing different design options. The light pen is used extensively as a natural tool for interaction between man and machine. The light pen is used to point to objects on the screen as well as to move the dies up and down in order to visualize how the preform fits in the finishing dies.

The CAM phase can be entered after preform designs are completed, or if a file of preform cross sections are available, it can be executed immediately; this allows slight modifications to the positions of the cross sections. Given a cutter diameter, the CAM phase blends the cross sections and calculates the cutter paths necessary to machine the preform surfaces.

Finally, an NC tape file is generated which can be punched onto paper tape and taken to an NC machine for manufacturing a model or an electrode.

It is possible to produce various intermediate preform designs (blocker designs) by using the preform design file as input to TRACKS. Appendixes II, VII, VIII, IX contain the details of the computer programs that make up TRACKS.

System Limitations

The system of computer programs, TRACKS, has a number of limitations which a potential user should be aware of. These limitations do not affect the application of TRACKS to track shoes and links; however, they will limit the immediate applicability of TRACKS to some other forgings. It is believed that most of these limitations can be removed by the addition and/or modification of appropriate subroutines. These known limitations include:

(1) At present, TRACKS cannot handle cross sections which exhibit plane-strain metal flow and which have a flash land on only one side. Two kinds of metal flow, axisymmetric (radial flow) and plane strain (parallel flow), cover most of the flow patterns in a given forging. Cross sections of a forging represent either axisymmetry or plane-strain metal flow. Each cross section can have a flash land on one side or on both sides. Thus, potentially, four types of cross sections are possible. At present, axisymmetric cross sections with flash on one side and on both sides, as well as plane-strain cross sections with flash on both sides can be handled by TRACKS. The ability to handle plane-strain cross sections with flash on one side could be added in the future, if required by the users.

(2) At present, only preforms for plane-strain cross sections and not those for axisymmetric sections can be manufactured by TRACKS. The manufacture of preform templates for round parts would involve the addition of a new subroutine.

(3) Ideally, TRACKS should be able to handle a mixture of plane strain and axisymmetric cross sections, be able to place them correctly in space, and calculate the blends in between and determine the cutter paths for NC machining. At present, TRACKS can only produce NC tapes for parallel plane-strain cross sections. To implement the ability to handle a mixture of cross sections would require considerable additional effort and may not be necessary in most practical cases.

APPLICATION OF "TRACKS" TO CAD/CAM OF TRACK-SHOE FORGINGS

The system of computer programs, called TRACKS, was tested by applying it to the T-130 track shoe. Cross sections of this track shoe were used as test data throughout the development of TRACKS. The upper and lower surfaces of T-130 are shown in Figures 5 and 6, respectively. The five cross sections (labeled 2-6), used in this application, are depicted by the dotted curves in Figures 5 and 6.

Data Preparation and Input

Each cross section of a forging is represented by a polygon. The polygon, in turn, is described the x,y,z coordinates and the associated radius at each vertex. These data are obtained from the finish forging drawing and tabulated by a draftsman. A disk file of the data for use by TRACKS is created using program ENTRDT (see Appendix IX). Alternative methods of creating this data file on the disk would be via punched cards or punched tape. The required format of the data file is detailed in Appendix VII.

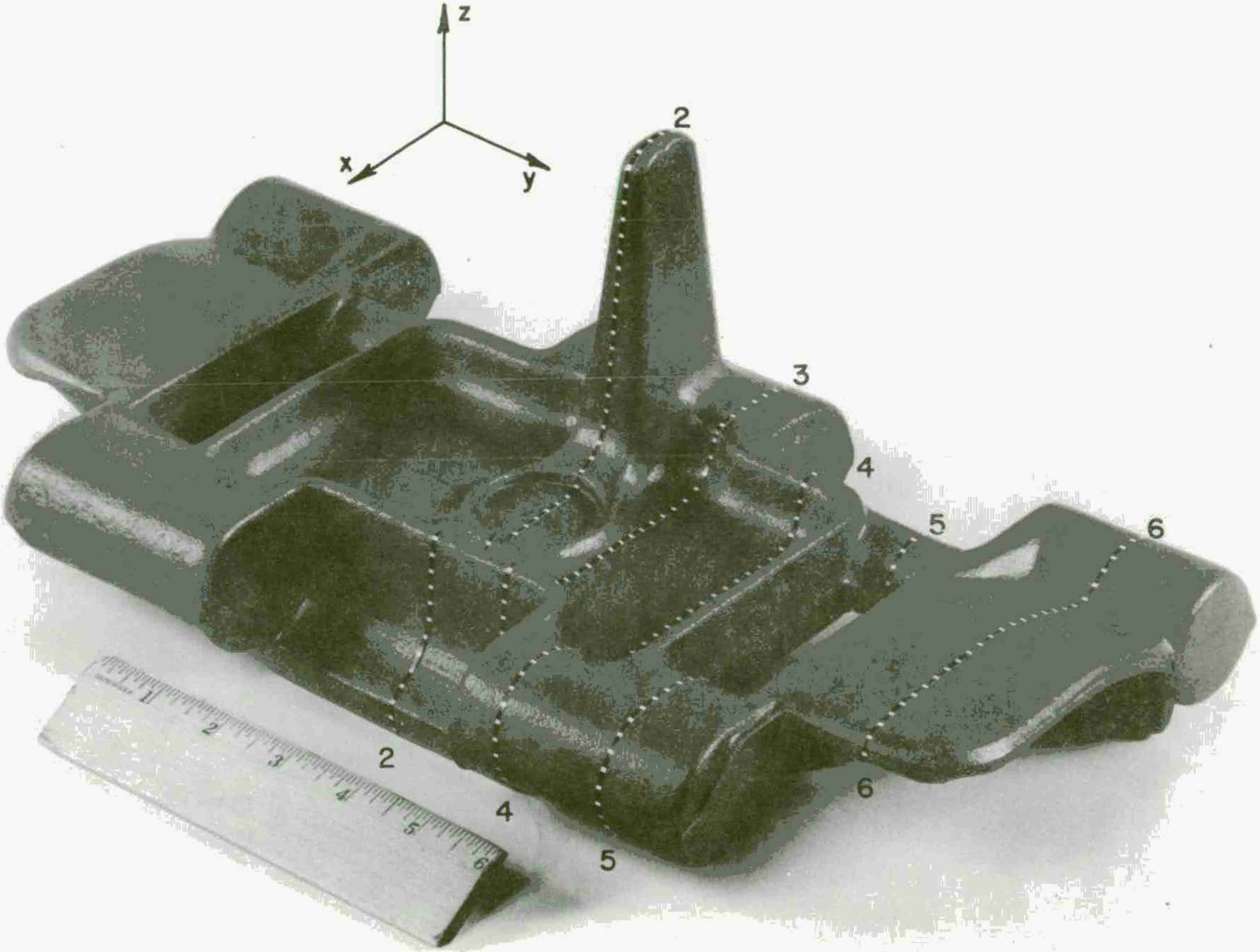


FIGURE 5. UPPER SURFACE OF T-130 TRACK-SHOE FORGING

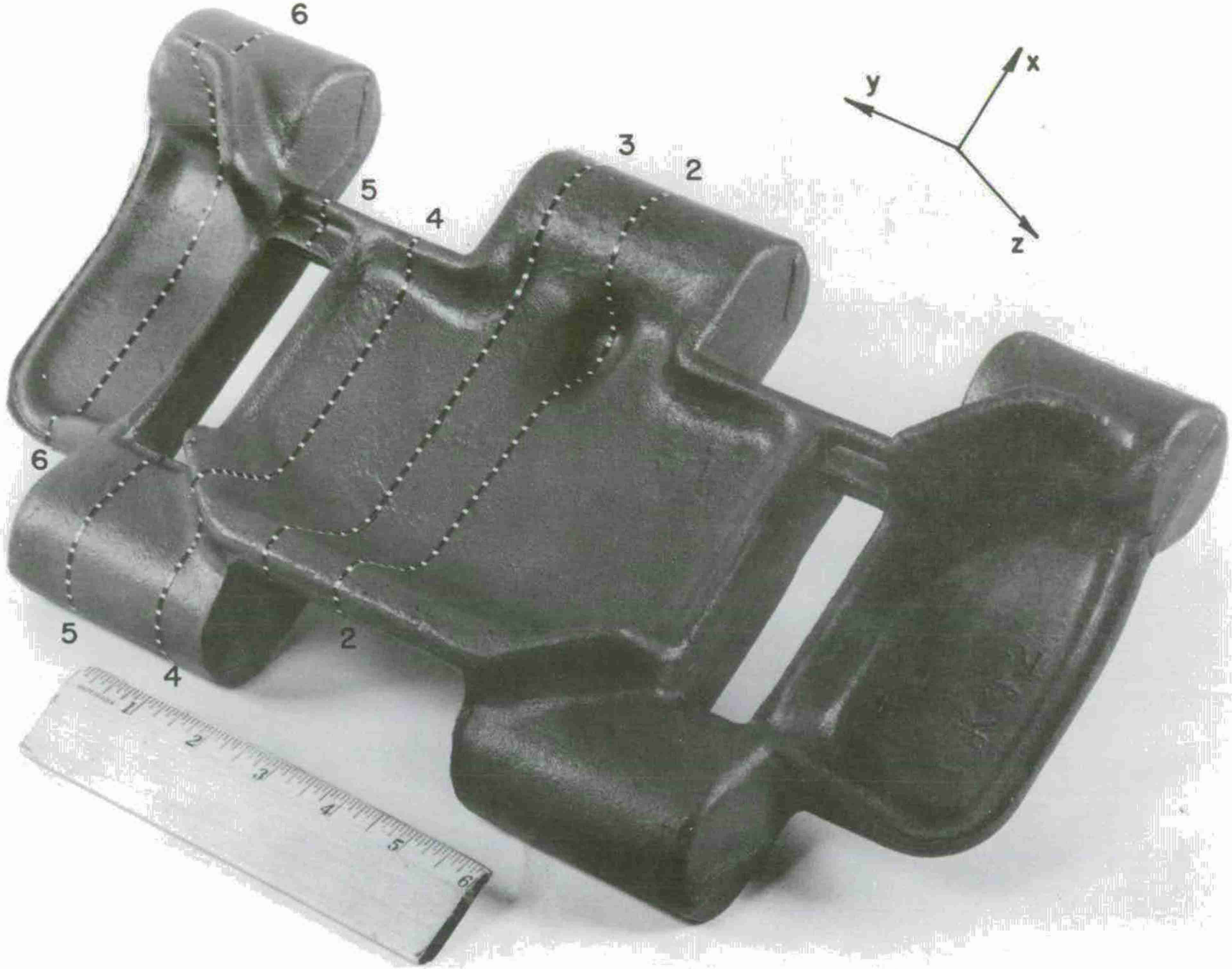


FIGURE 6. LOWER SURFACE OF T-130 TRACK-SHOE FORGING

Load and Stress Calculations

Load and stress calculations were performed using the cross-sectional data for the T-130 track shoe, as seen in Figures 5 and 6. In this procedure, the stress distribution and the average pressure are determined for each cross section. Using the "depth" of a cross section, the forging load acting on that cross section is calculated. The sum of all these loads per cross section gives the total forging load.

At the start of stress calculations, each cross section is displayed to the user at the cathode ray tube (CRT). The user is asked by the computer to point to the cavities, or ribs, of the cross section so that a metal flow model can be determined by the computer program. Figure 2 illustrates the flow model with the "shear surfaces" along which metal flows after the die cavity and all the ribs are filled. The user is also asked to input (a) the friction factor, f, (b) the flow stress, $\bar{\sigma}$, of the forged material under the particular forging conditions, and (c) the flash dimensions.

The friction shear stress, τ, at the die-forging boundary is expressed as $\tau = f\,\bar{\sigma}$. The friction factor, f, is usually determined by conducting a ring test under forging and lubrication conditions, similar to those encountered in production(1,2). In mechanical press forging of steel components with graphite-based lubricants, f has been found to be between 0.2 and 0.3(2). In estimating the stresses and the load for forging the T-130 track shoe, the friction factor, f, was estimated to be f = 0.25, at midrange of values measured for practical conditions.

The flow stress, $\bar{\sigma}$, of the forging material is primarily influenced by the rate of forging, i.e., strain rate, $\dot{\varepsilon}$, and the forging temperature, θ. In the present program, the flow stress, $\bar{\sigma}$, was estimated from experimental data available from previous studies(2). The T-130 track shoe is forged from AISI 4140 steel. There are no published flow-stress data at forging temperatures available for this alloy. However, under forging conditions, the flow stress of this material is close to that of AISI 1045 on which flow-stress data are available. The maximum forging load is measured in the finish forging stage, which is the third forging station in track shoe forging with a mechanical press. The first two stations are preblocking and blocking, or preforming. In finish forging, the stock temperature can be expected to be about 2000 F. The amount of deformation, or strain, $\bar{\varepsilon}$, is relatively small and is in the order of $\bar{\varepsilon} = 1$. The strain rate, $\dot{\bar{\varepsilon}}$, is between 16/sec to 20/sec, as measured in earlier investigations(2). Thus, for AISI 1045, using $\theta = 2000$ F, $\bar{\varepsilon} = 1$, and $\dot{\bar{\varepsilon}} = 18$, the flow stress, $\bar{\sigma}$, is estimated from the data, given in Reference 2, to be $\bar{\sigma} = 15{,}000$ psi.

Using $f = 0.25$, $\bar{\sigma} = 15{,}000$ psi at 2000 F average forging temperature, the computer programs calculate 3500 tons as the total load required to forge the T-130 track shoe. This figure is based on a flash geometry of flash width = 0.250 inch, flash thickness - 0.125 inch. This track shoe is forged in a 4000-ton Erie mechanical press(7). Thus, it appears that the calculated load values are well within acceptable engineering tolerances.

Estimation of Forging Energy

The energy required to produce a forging is the area under the load-stroke curve, as illustrated in Figure 7. The knowledge of the required forging energy is especially critical in mechanical-press forging, as it is practiced in producing track shoes. If the energy required by the forging process equals or exceeds that available in the mechanical press, then the press will slow down by an unacceptable amount, thereby limiting the production rate.

In order to estimate the forging energy, it is necessary to calculate the forging load at various stroke positions. Due to complexity of the parts considered in this program, calculating the load at each stroke position is an extremely complicated task. Therefore, an approximate method for energy calculations was developed. The details of this method are given in Appendix V.

Briefly, the track-shoe forging is considered as a simple rectangular part with two round ends, as seen in Figure V-2 of Appendix V. The cavity and flash dimensions of this "simplified" forging are calculated such that this forging simulates, fairly closely, the actual track-shoe forging. Thus, the load-stroke curves of both "simplified" and actual forgings are expected to be similar so that the areas under the load-stroke curves, i.e., the energy in inch-tons, will be approximately the same in both actual and simplified cases.

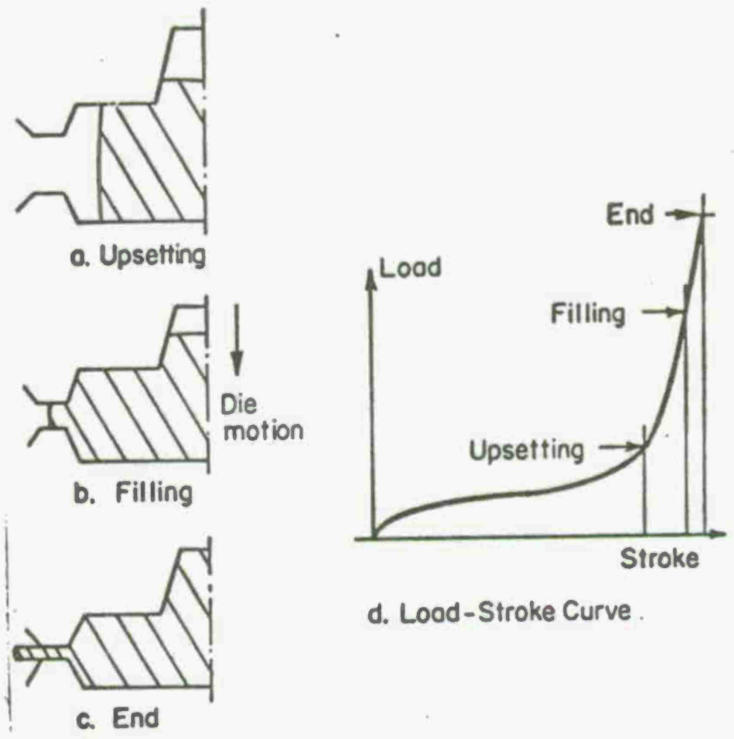


FIGURE 7. SCHEMATIC ILLUSTRATION OF METAL FLOW AND LOAD-STROKE CURVE IN CLOSED-DIE FORGING

Using this approach, the energy necessary to forge the T-130 track shoe from a round-cornered square forging stock is estimated to be about 5400 inch-tons. Considering that the maximum forging load is estimated to be about 3500 tons, the calculated value of the energy appears to be quite reasonable.

Preform Design

After calculating the load on each finish section, each section was modified to create the preform geometry. Guidance of general nature for this design process was provided by the design engineers of FMC's Steel Products Division. In general, many of the small details of the finish forging cross section were eliminated and radii were enlarged to permit easier metal flow from the billet into the preform die. The finish and preform geometries for Sections 2-2 and 4-4 are shown in Figures 8 and 9. The other sections were modified in a similar manner.

The preform design is constructed on the cathode ray tube (CRT) screen by interacting with the computer via the light pen and the keyboard. The features of the finish-die geometry is used to create the preform die geometry. The computer is used to perform all the necessary arithmetic; the designer performs all the decisions that require visual perception.

Figure 9 illustrates the capability of "opening and closing" the dies on the screen. The user can "move" the upper and lower dies, as well as the preform, on the screen using the light pen. This gives the designer the ability to visualize how his preform fits into the finish dies, especially the location of the initial contact points (pickup points) which have a bearing on expected die fill.

As the preform design for each cross section is completed, the geometry is saved on a disk file for subsequent use by the CAM section. The details of the preform design procedure are given in Appendixes I and VII.

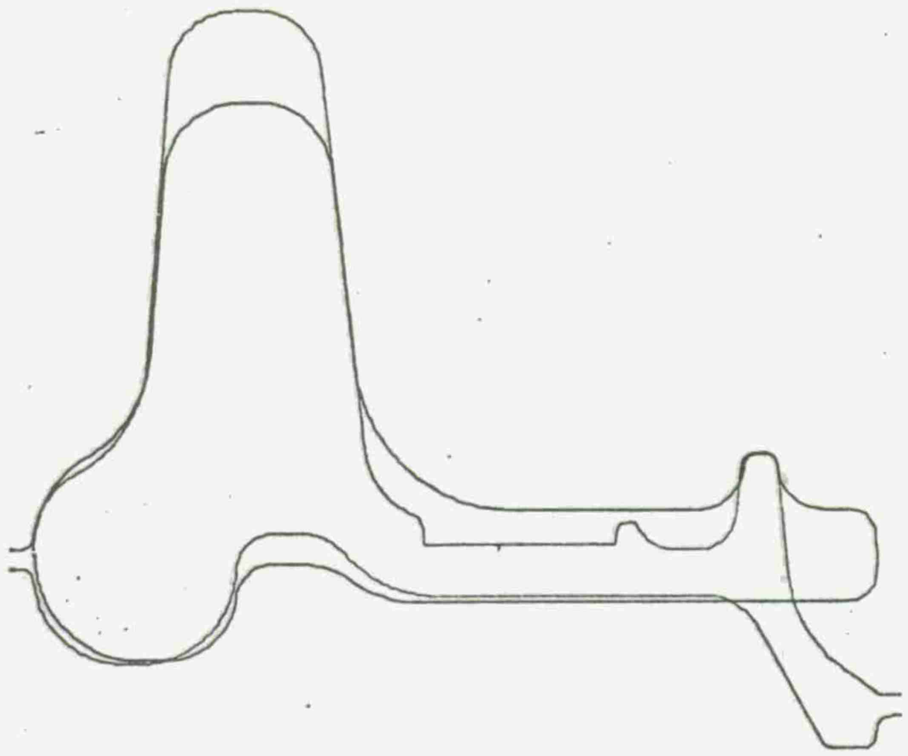


FIGURE 8. T-130 TRACK SHOE, SECTION 2-2. PREFORM SECTION AND FINISH DIES (Dies Closed)

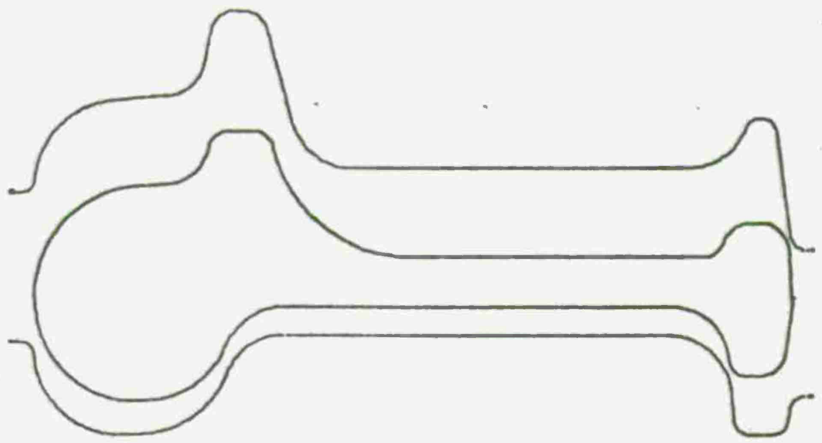


FIGURE 9. T-130 TRACK SHOE, SECTION 4-4. PREFORM SECTION AND FINISH DIES (Dies Open)

NC Machining of the Preform Surface

After the designs of the preform cross sections are completed, the CAM section of TRACKS is entered. The major input to this section is the file created by the preform design section. In addition, the desired draft angle between adjacent cross sections and the cutter diameter are also input.

The preform surface was generated, based on a 7-degree draft between adjacent sections, and a 2-inch vertical feed at the start and end of the cut.

Plots of the upper and lower cutter center-line paths are shown in Figures 10 and 11. Figures 12 and 13 show the wood preform models which were produced. Each model was made in two steps. They were first rough machined using an 0.500-inch diameter ball mill, and then finished with an 0.250-inch diameter ball mill. Separate tapes for each cutter size and each surface were used. The tapes contained from 33,280 characters, for the shortest tape, to 41,500 characters, for the longest tape. This is equivalent to a length range of 280 ft to 350 ft.

In Figure 12, the photograph of the upper surface preform model, a block can be seen behind the large "horn" projection. This was added to prevent the horn from breaking off from the main body while being machined. It should also be noted that NC tapes were made for only half the track shoe since the track shoe is symmetrical around the X-Z plane. The other half of the preform model can be machined by using the mirror image function of the numerical controller on the NC milling machine. Thus, Figures 10, 11, 12 and 13 are only for one-half of the blocker for the T-130 track shoe.

Application to Other Track Shoes and Other Forgings

In addition to the in-depth evaluation of TRACKS, using the five sections from the T-130 shoe, three sections from a T-142 shoe were also tested. A current supplier of the T-142 was not contacted, so the quantitative results were not evaluated. However, all functions and capabilities for load analysis, preform design, and surface generation operated properly. The

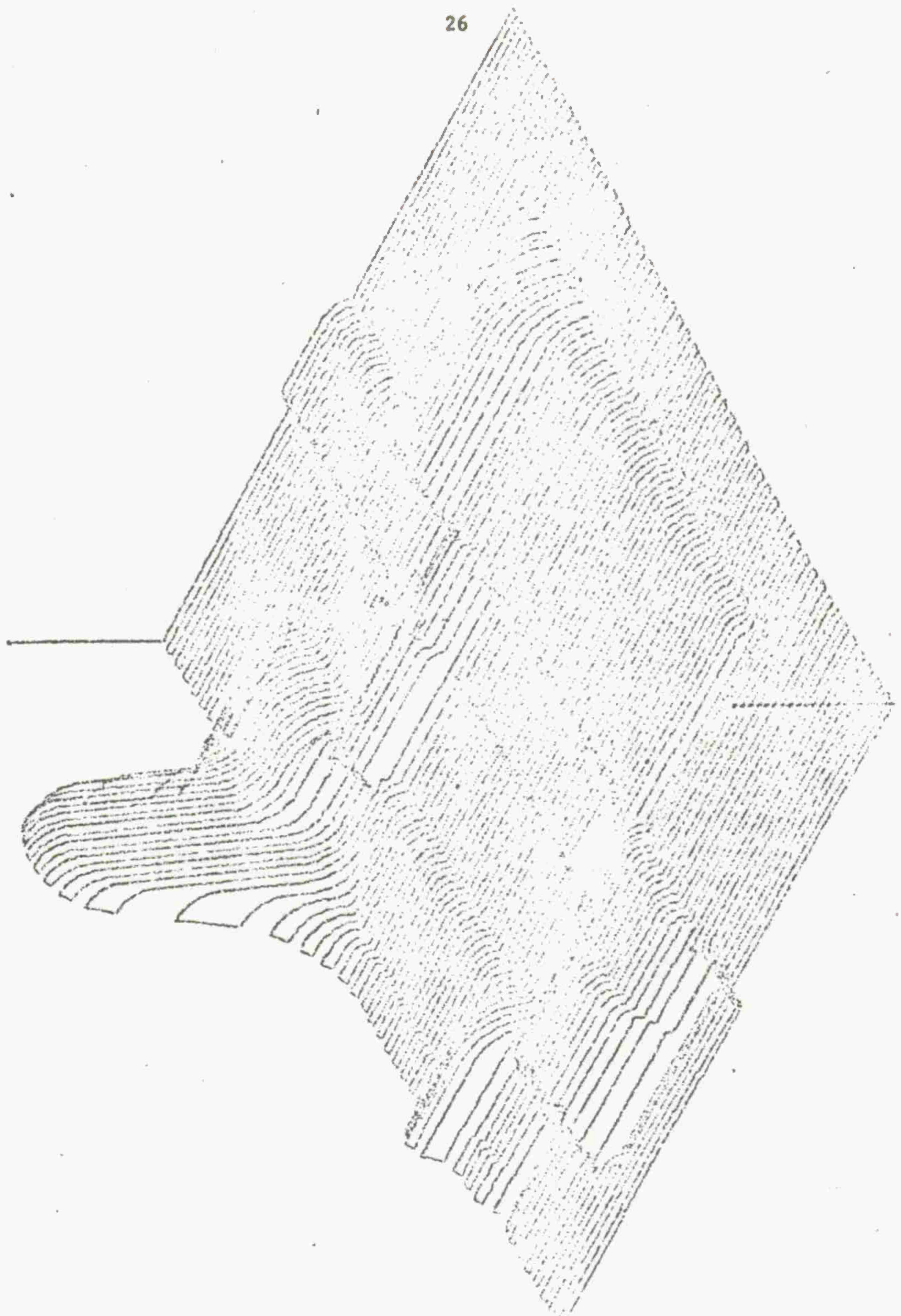


FIGURE 10. T-130 TRACK SHOE PREFORM CUTTER PATHS FOR 0.5-INCH BALL MILL (Upper Surface)

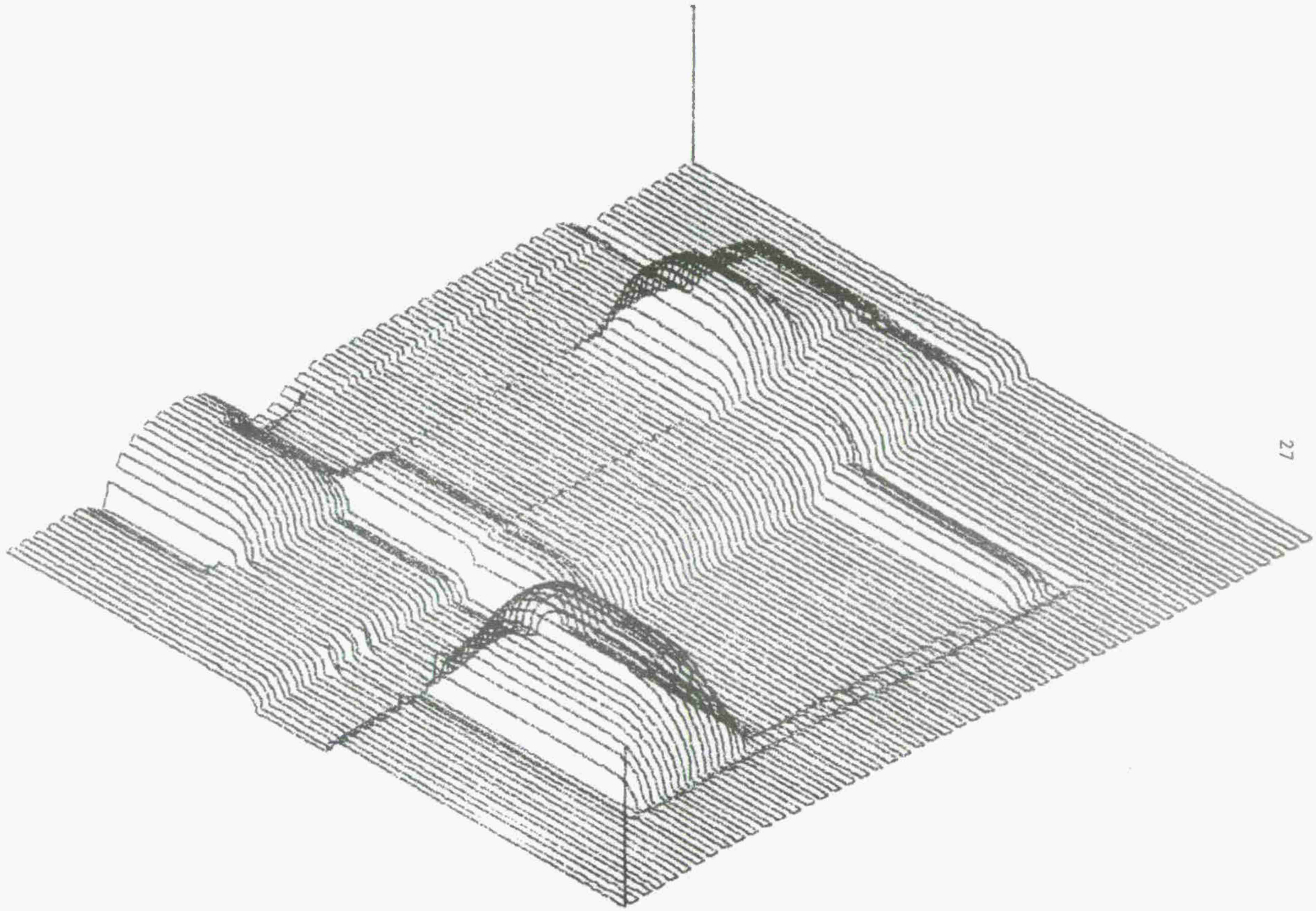


FIGURE 11. T-130 TRACK SHOE PREFORM CUTTER PATHS FOR 0.5-INCH BALL MILL (Lower Surface)

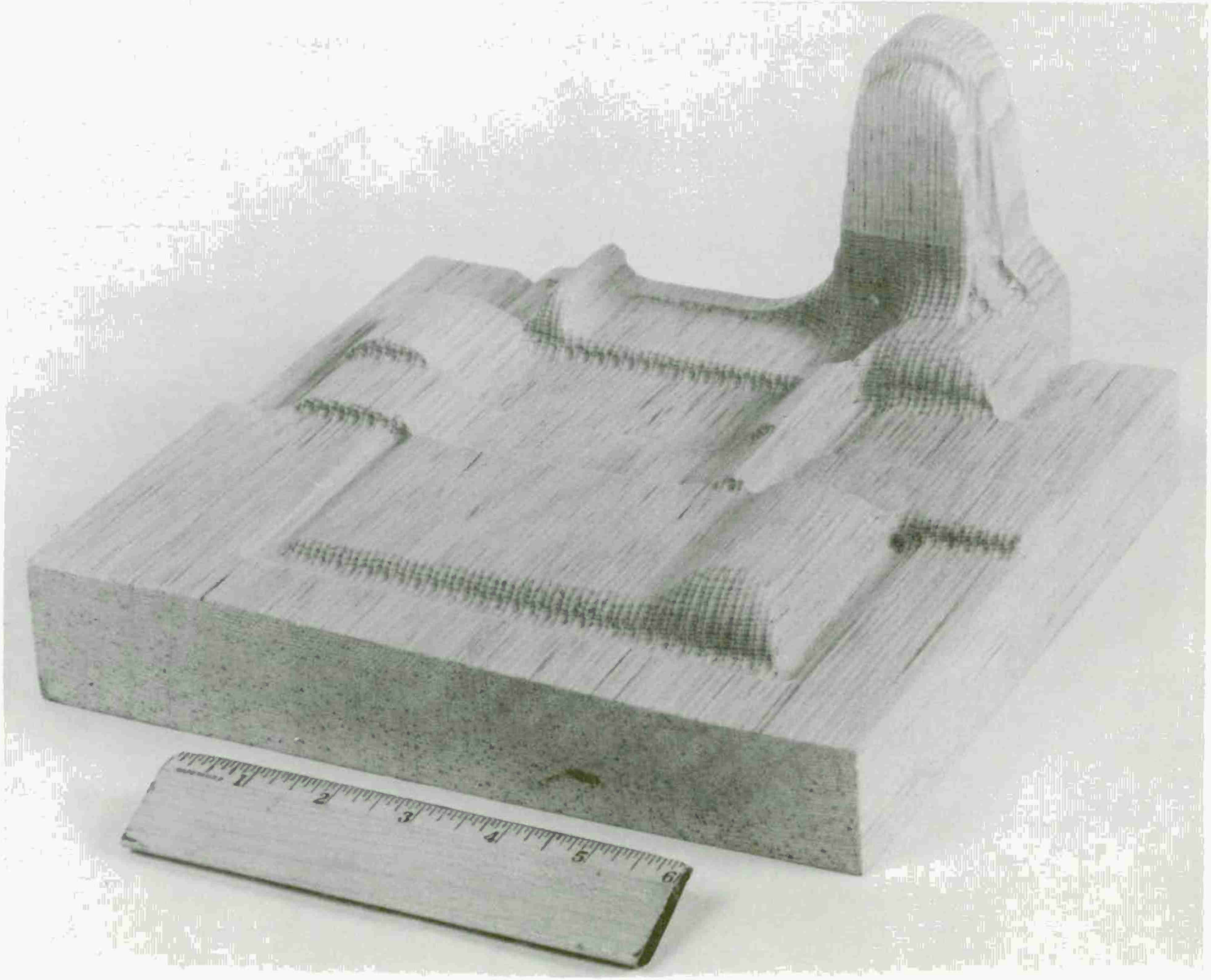


FIGURE 12. NC MACHINED WOOD MODEL OF T-130 TRACK-SHOE UPPER SURFACE PREFORM

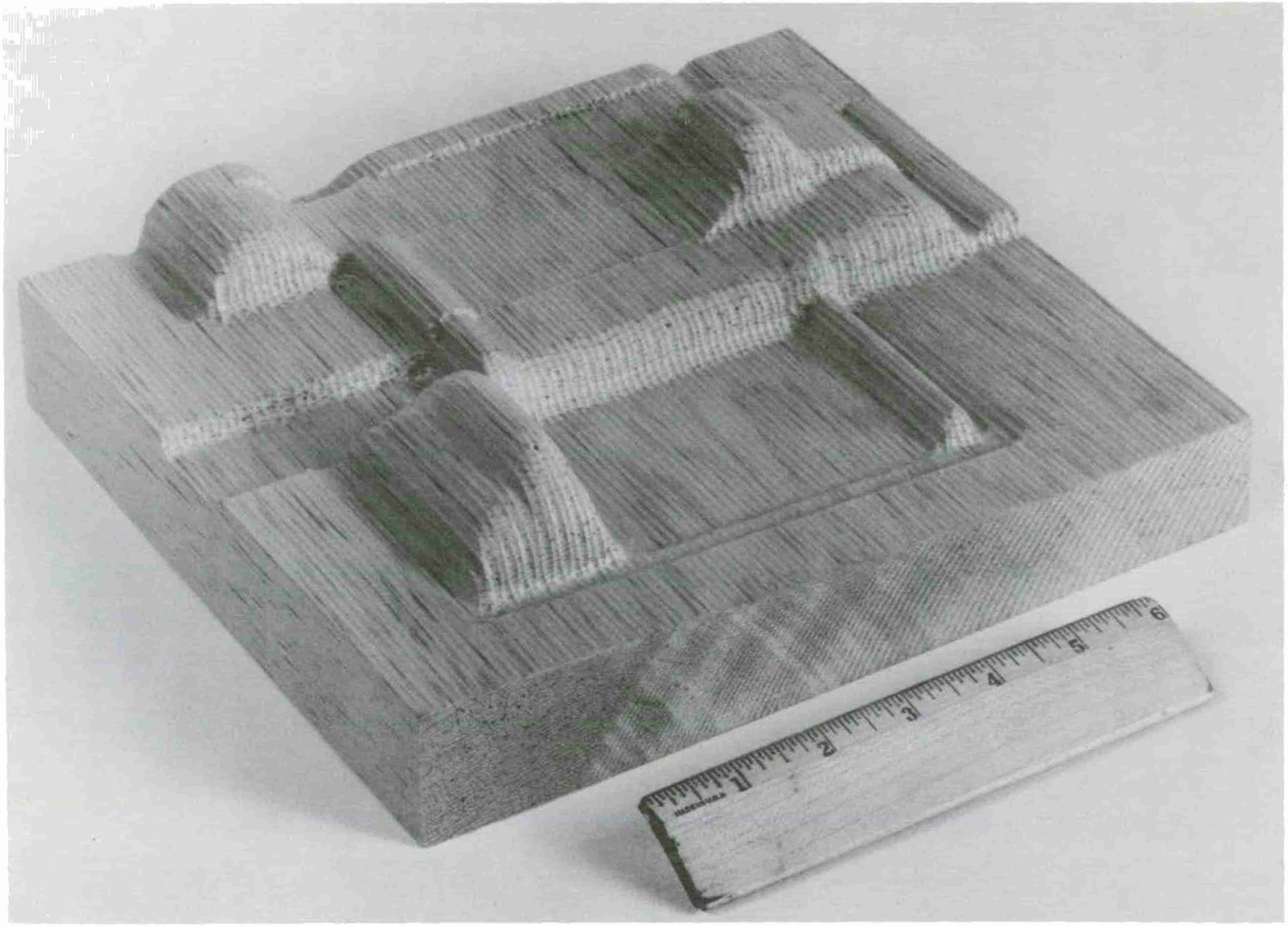


FIGURE 13. NC MACHINED WOOD MODEL OF T-130 TRACK SHOE, LOWER SURFACE PREFORM

expected forging load was calculated to be about 4150 tons, assuming the same forging conditions as for T-130. This track shoe has a smaller plan area than T-130; however, as seen in Figure 14, overall it is a much thinner part, thus requiring the higher load.

Although TRACKS was developed for track shoes and links, it can be used for most any forging application.

SUMMARY AND CONCLUSIONS

In this project, a system of computer programs for computer-aided design and numerically-controlled machining of preform forging dies was developed. This system is implemented as a stand-alone system on a minicomputer. It is capable of calculating the expected forging load, the stock volume required, the plan area and average forging pressure. In addition, the system is capable of designing preform cross sections via interactive graphics and determining the cutter paths for NC machining blocker dies.

Although the system was developed around a track shoe, it is completely general in its applicability. Track shoes do not exhibit any geometric modularity; therefore, the computer programs were written so as to be able to handle most any cross-sectional geometry. Thus, not only track shoes and links, but also crank arms, gear blanks and other forgings could be processed through TRACKS.

Load and energy calculations are useful not only in forging press selection, but also as an indirect guide to estimate die life. Stress calculations can guide the designer in determining the optimum flash dimensions. Although restrictive flash geometry is desirable to promote die fill, it also results in high die stresses. The stress distribution determined by TRACKS could be used as a guide for preventing die breakage. In the ideal case, the same stress distribution would be needed as part of the input (boundary condition) to computer programs for analyzing the stress state within the die in order to prevent premature die failure.

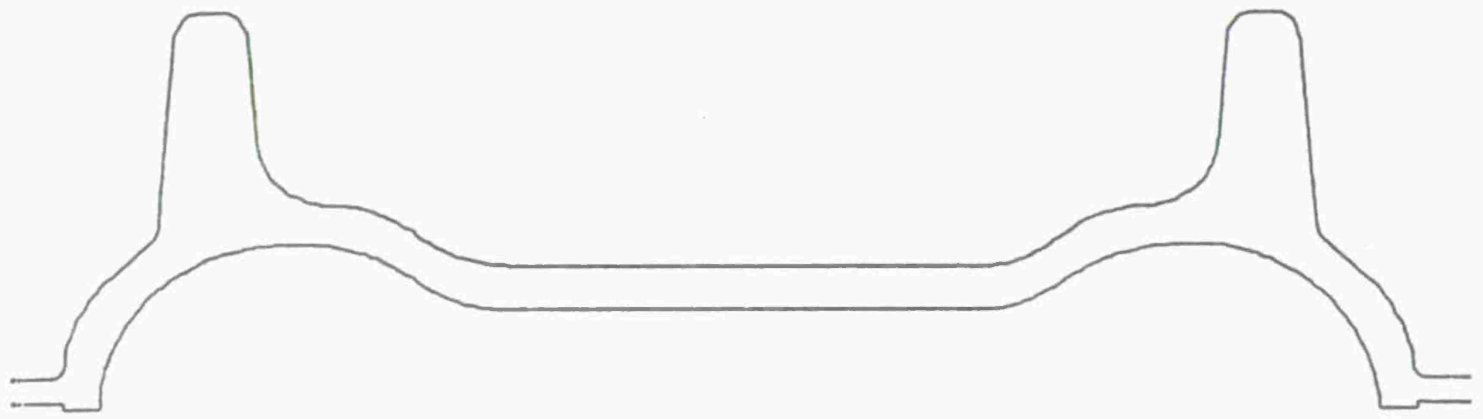


FIGURE 14. CENTRAL CROSS SECTION OF T-142 TRACK SHOE

Our survey of preform design techniques showed that there are some general guidelines, but no one method that is applicable to all forgings. Accordingly, preform design must utilize the experience of the designer. CAD techniques speed up the calculation and drafting portion of preform design and they enhance accuracy and reproducibility. Since manipulation of geometry is the major activity of the design process, interactive graphics is the most natural way of establishing communication between the computer and the designer.

NC machining of preform appears to be economically attractive since the geometry is already in computer readable form. Also, the geometry is truly defined only on the cross sections. The surface in between the cross sections is left to the die maker to "blend in". Approximately 80-90 percent of this blending can be done satisfactorily by computer programs and by NC machining. This approach speeds up the die manufacturing process and ensures high reproducibility in die making. After NC machining of most of the preform surface, the 10 to 20 percent of the metal removal that is left is best done by hand.

The system of computer programs, TRACKS, should be considered as only a starting point in updating existing methods of forging die design and manufacture. Additional work needs to be done in implementing such a system, in evaluating the results, and for modifying the system as needed so that CAD/CAM application in forging becomes a routine procedure, used daily under production conditions.

REFERENCES

(1) Altan, T., et al., "A Study of the Mechanics of Closed-Die Forging, Phase I", Final Report prepared by Battelle's Columbus Laboratories for AMMRC, Watertown, Massachusetts, September, 1970, Contract No. DAAG46-68-C-0111, AD711-544.

(2) Douglas, R. J., and Altan, T., "A Study of Mechanics of Closed-Die Forging, Phase II", Final Report prepared by Battelle's Columbus Laboratories for AMMRC, Watertown, Massachusetts, November, 1972, Contract No. DAAG46-71-C-0095.

(3) Akgerman, N., and Altan, T., "Application of CAD/CAM in Forging Turbine and Compressor Blades", Transactions ASME, April, 1976, pp 290-296.

(4) Voigtlander, O., "The Manufacturing of Blades for Turbines and Compressors: Precision Forging of the Airfoil", (in German), Industrie-Anzeiger, Vol. 91, No. 40, May 13, 1976, pp 908-913.

(5) Anonymous, "Computerized Forging", Manufacturing Engineering Management, February, 1974, p 19.

(6) Akgerman, N., Subramanian, T. L., and Altan, T., "Manufacturing Methods for a Computerized Forging Process for High-Strength Materials", Final Report prepared by Battelle's Columbus Laboratories for Air Force Materials Laboratory, Wright-Patterson Air Force Base, Ohio, January, 1974, Contract No. DAAG46-68-C-0111.

(7) (Advertisement) Production, October, 1972.

APPENDIX I

OPERATION AND USE OF TRACKS CAD/CAM SYSTEM

APPENDIX I

OPERATION AND USE OF TRACKS CAD/CAM SYSTEM

TRACKS is a system of programs which is intended to handle a general class of closed-die forgings typified by track shoes for crawler vehicles. It develops the parameters (area, perimeter, center of gravity, etc.) for given cross sections, and performs stress analysis and preform (blocker) design. The following conventions are used in structuring the system:

(1) ANSI FORTRAN IV standards are followed as much as it is practical.

(2) A cartesian coordinate system is used with each cross section being in a plane parallel to the direction of ram motion, indicated by the Z axis. For the purpose of analysis, the plane of each cross section is rotated to the Z-X plane of the screen of the cathode ray tube (CRT), or to that of the paper, as illustrated in Figure I-1.

(3) The coordinate points of the polygon, which represents a given cross section, are assumed to be in clockwise order. Such a polygon is shown in Figure I-2, for the center cross section of the T-130 track shoe. The first point of the polygon is located as follows, Figure I-3:

(a) For plane-strain sections, point 1 is the left-hand parting line (i.e., it has the minimum X value when viewed normally to the section).

(b) For axisymmetric sections with single flash (i.e., pie-shaped sections of a solid disk), the axis of symmetry is on the left side of the part, and point 1 is the midpoint along the axis between the upper and lower surfaces.

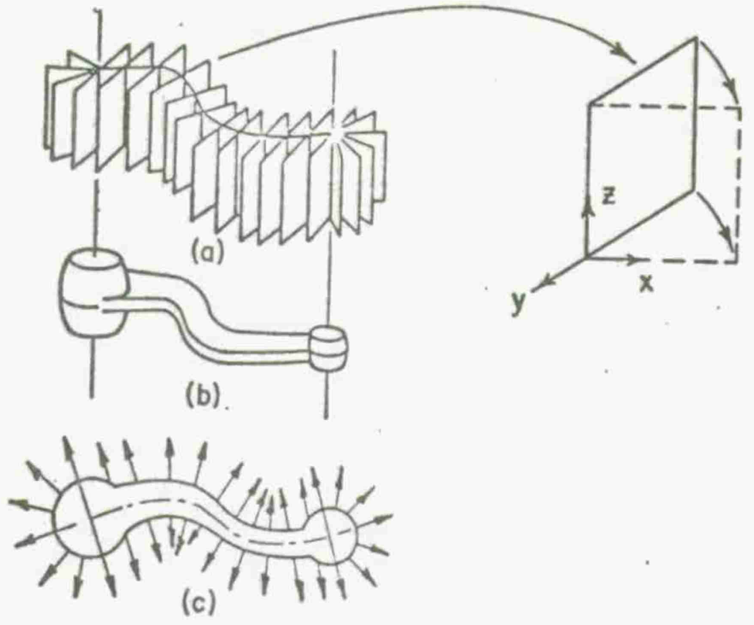


FIGURE I-1. CROSS SECTIONS OF A FORGING AND ROTATION OF A CROSS-SECTIONAL PLANE FOR ANALYSIS

(a) Cross Sections or Planes of Flow
(b) Finish-Forged Shape
(c) Directions of Metal Flow

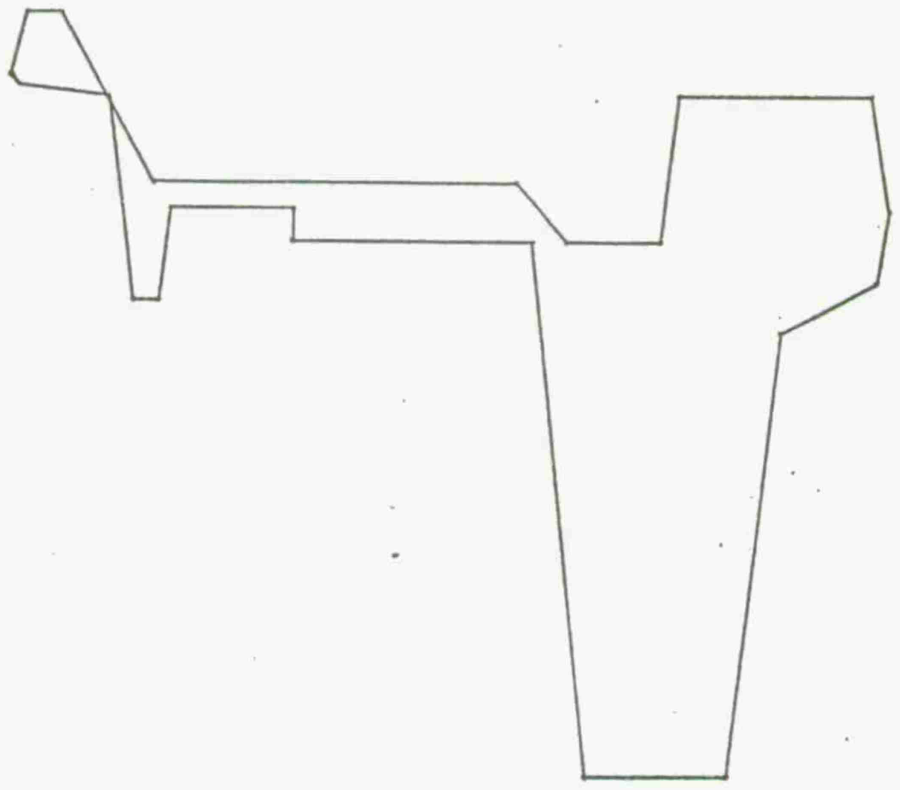
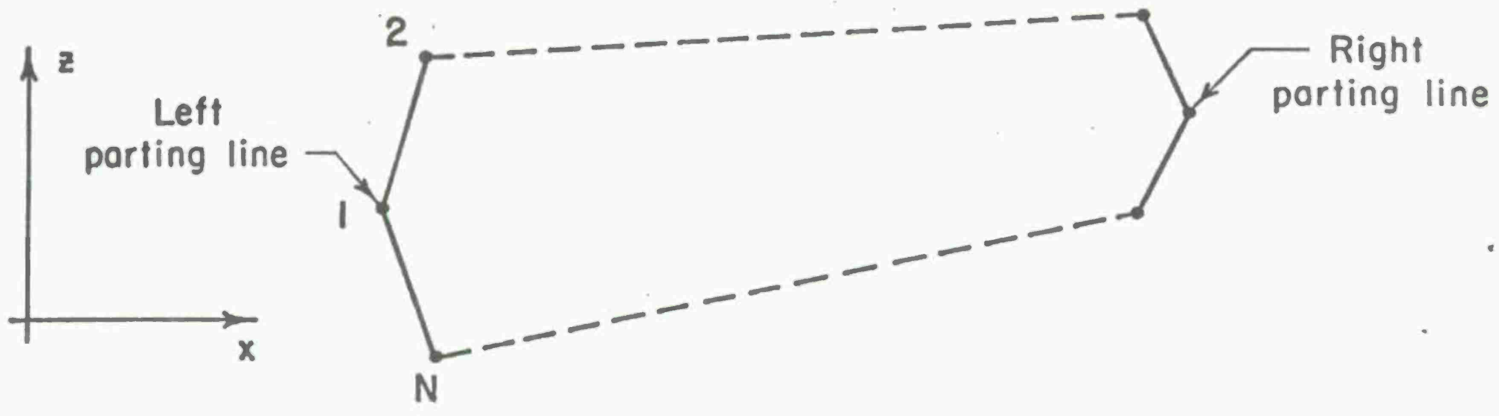
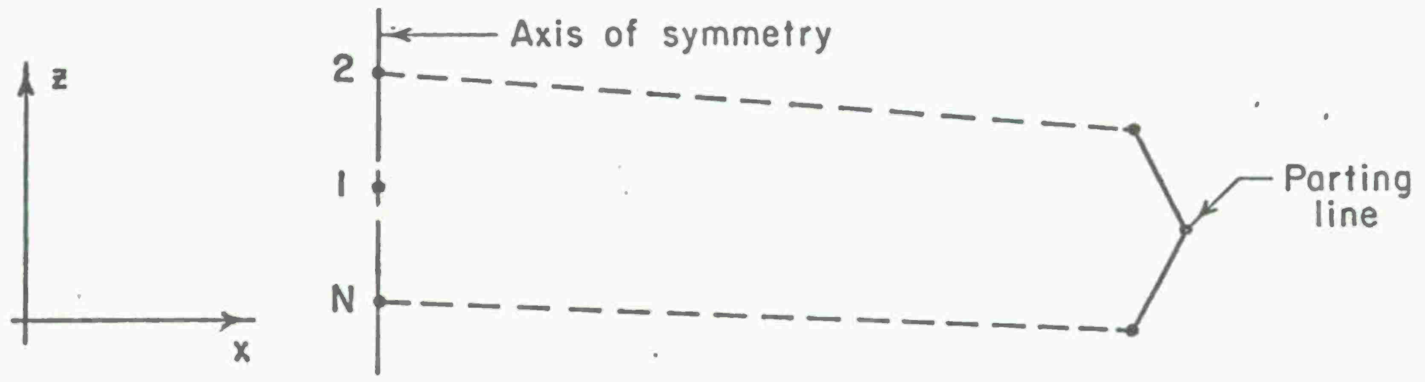


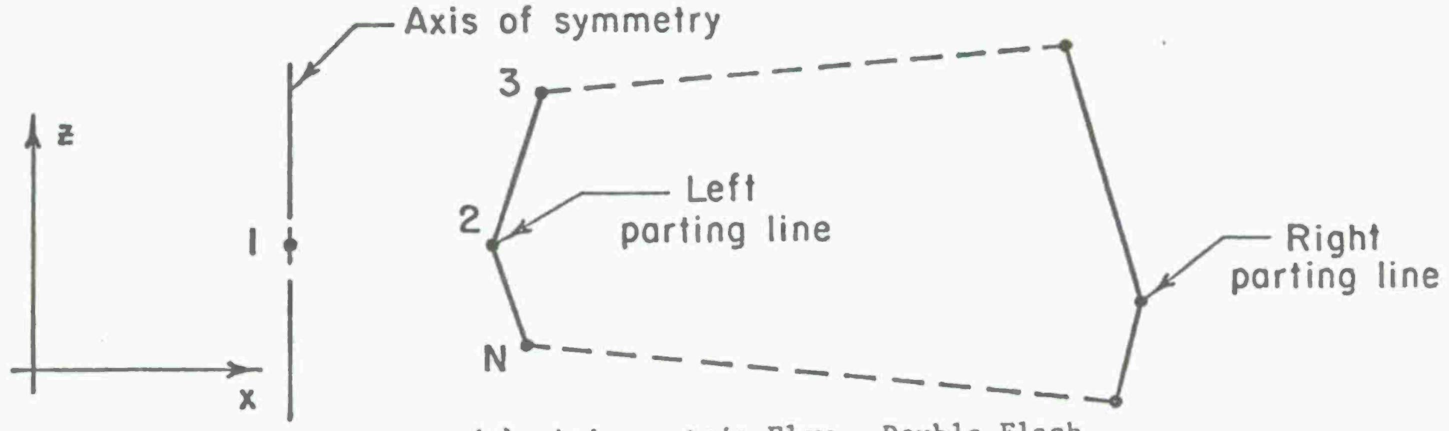
FIGURE I-2. ORIGINAL INPUT POLYGON FOR THE CENTER CROSS SECTION OF THE T-130 TRACK SHOE



(a) Plane-Strain Flow - Flash Both Sides



(b) Axisymmetric Flow - Single Flash



(c) Axisymmetric Flow - Double Flash

FIGURE I-3. COORDINATE NUMBERING CONVENTION USED IN DESCRIBING A FORGING CROSS SECTION IN THE FORM OF A POLYGON

(c) For axisymmetric sections with double flash, (i.e., pie-shaped sections of a disk with a hole in the center), the axis of symmetry is to the left of the section. The data are input with point 1 being the X-Z coordinate of the axis of symmetry. The location of the points are altered within the program by saving point 1 in a special location, and then shifting all points backwards 1 location in their arrays.

(4) When flash occurs on both sides, the minimum value of X will determine the left-hand parting line. In all cases, the maximum value of X will determine the right-hand parting line.

(5) No more than two consecutive corners (negative radii) are permitted. A rib may exist if a fillet/corner/corner/fillet condition exists in this sequence.

(6) The program is intended to operate interactively with a designer, using a Cathode Ray Tube (CRT) display terminal. The program is not designed to run in a batch mode of operation.

The general procedure for analyzing and designing die sections, and for developing the NC cutter data for the corresponding preform is shown in Figure I-4. When started, TRACKS asks the operator to specify what is to be done -- CAD(1) or CAM(2). By typing the appropriate numerical response, the corresponding section is entered.

When the CAD (design) phase is requested, TRACKS asks the user to specify the various data files to be used or generated, as follows:

(1) Print File -- used to store all results generated by TRACKS for subsequent off-line printing. By having results sent to this file, the hard-copy portion of the terminal may be disabled. The user then inputs data via the terminal keyboard, and receives output on the CRT screen.

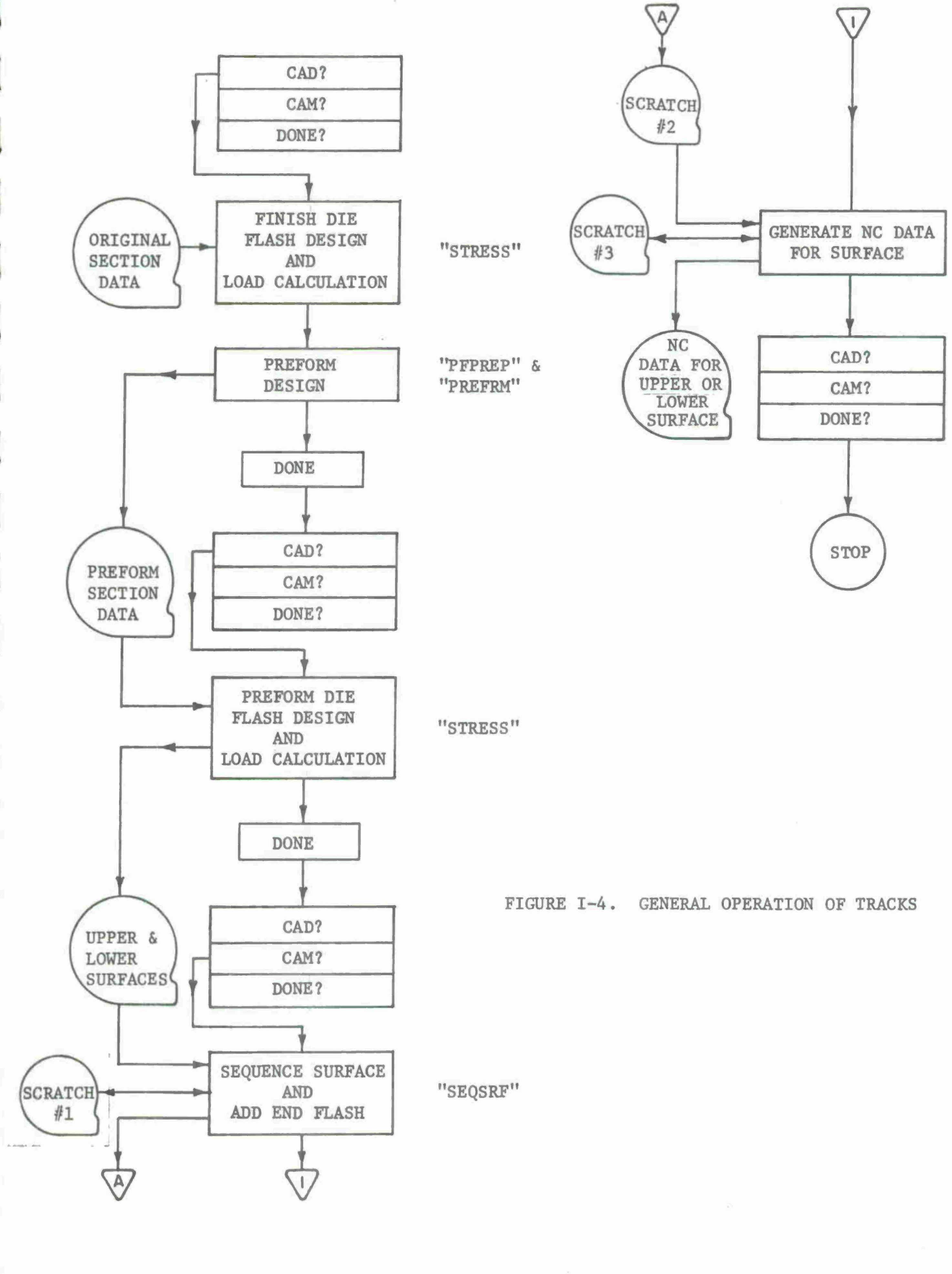


FIGURE I-4. GENERAL OPERATION OF TRACKS

(2) Data File -- the file where the input data is stored.

(3) Blocker Polygon File -- the file used to store the data for the blocker (preform) polygon. This data is a result of the modifications made to the finish part polygon by the user during the design process.

(4) NC Prep File -- the file used to store the interpolated data for the upper and lower die surfaces. This data is used in the CAM phase to generate the cutter paths required to machine the preform models.

After the appropriate file names are entered, TRACKS asks for the gutter limits. The gutter limits define the range, along the X-axis, of the block from which the preform model will be machined. The minimum gutter limit can be taken as the smallest X-coordinate of any cross section less one inch; the maximum gutter limit is the largest X-coordinate of any cross section plus one inch.

TRACKS next asks for the number of the section to be processed. If no value is entered, it proceeds to operate on the data for the next section in the data file. If a positive value is entered, a search is made of the data file until a match is found between the value entered and a section with the same number in the file. If a negative value is entered, the data file is first rewound and then a match searched for. At start up, entering a negative value has no effect since the data file is always opened in a rewound condition.

Flash parameters are requested next. If no value is entered, the default value for each is used. The operator is then asked to indicate if he wishes to have intermediate results saved. They will be saved only if he enters "Y" (Yes). The intermediate results are primarily intended as a diagnostic aid, to be used in case a program bug is encountered.

TRACKS then calls subroutine PRPROS to obtain the data for the section specified. After finding and reading the data, preliminary analysis of the section geometry is performed and the results are written on the CRT. If this preprocessing results in any errors, ERRPRT is used to report the problem. Depending on the nature of the error, TRACKS will either attempt to continue or terminate the design process, after giving the appropriate messages to the user.

Figure I-5 is a listing of the man/machine dialog used when starting TRACKS. Underlined text was entered by the designer. Where no response is shown, the default values were used.

If no errors are encountered in PRPROS, PICTUR is used to (1) add the flash dimensions to the cross section, (2) generate the small, straight-line segments used to represent arcs on the CRT, (3) initialize the graphics handler, and (4) draw the picture of the part. When PICTUR is completed, TRACKS asks the designer to indicate what operation he wishes to perform next. The options available are:

- Next Section (1)
- New Flash (2)
- Stress (3)
- Preform (4)
- Done (5).

The number in parentheses indicates the next design step to be selected by the designer. For example, by typing "3", the designer would initiate the stress analysis procedure. TRACKS evaluates the designer's response and branches accordingly.

If option 1, Next Section, is requested, TRACKS loops and asks for the new section number and flash parameters. It then proceeds as described above to find the proper section data and make the preliminary geometric analysis. If option 2, New Flash, is requested, the designer is asked to enter the new flash parameters. TRKFRG then recomputes the geometric parameters based on the new flash values for the current section being considered.

If stress analysis is requested, FLWSRF is started. This subroutine first asks the designer to enter the coefficient of friction at the material-die interface. This value depends on the die and billet materials, and the type of lubricant being used. Its value generally lines in the range of 0.20 to 0.40. The user is then asked if new shear boundaries are to be used. If any response other than "Y" (Yes) is entered, the previous boundaries indicated by the user will be re-used. This feature allows a section to be analyzed for several different flash geometries with the shear boundaries being indicated the first time only.

```
•R TRACKS

CAD(1), CAM(2), DONE(3)? 1

PRINT FILE NAME? *T13ØPR.LST

DATA FILE NAME? *T13ØSH.DAT

BLOCKER POLYGON FILE NAME? *T13ØBL.DAT

NC PREP FILE NAME? *T13ØNC.DAT

CUTTER LIMITS? (MIN & MAX) -1.5, 3.5

SECTION NUMER?


FLASH PARAMETERS

     WIDTH (Ø.25Ø) ?

     THICKNESS (Ø.125) ?

     RADIUS (Ø.125) ?

     FACTOR (4.Ø) ?


SEND DETAILS TO PRINTER? (Y/N)



SECTION NO.    2 IS IN PLANE STRAIN WITH DEPTH =  Ø.25 UNITS.

FLASH PARAMETERS:
WIDTH = Ø.25Ø,  THICKNESS = Ø.125,  RADIUS = Ø.125, FLASH FACTOR =    4.

CROSS-SECTION AREA =    9.39  PLAN AREA =    2.5Ø  PERIMETER =   23.73
PART VOLUME =    3.46  FLASH VOLUME =    Ø.Ø9  TOTAL VOLUME =     3.55
COORDINATES OF CENTROID ARE X =    2.24      Y =    Ø.ØØ


NEXT SECTION(1), NEW FLASH(2), STRESS(3), PREFORM(4), DONE(5)? 3
```

FIGURE I-5. TRKFRG OPENING DIALOG SEQUENCE. OPERATOR'S RESPONSES ARE UNDERLINED. WHERE NO RESPONSE IS SHOWN, DEFAULT VALUES ARE USED.

If new shear boundaries are to be used, TRACKS then gives instructions as to how cavity boundaries are to be indicated using the light pen. The upper and lower die surfaces are made light-pen sensitive, and two light-pen sensitive text strings, "ACCEPT" and "END", are displayed. When the operator touches the light pen to one of the surfaces, a small cross (+) is displayed on the surface at the location of the light pen. If the light pen is moved along the surface, the cross will follow the light pen. When the pen is lifted from the surface, the cross remains where it was last positioned. The operator is expected to use the light pen to indicate cavity boundaries across which metal flow in shear will occur. After positioning the cross on a cavity boundary, the designer then touches the pen to the "ACCEPT" text. When he next touches one of the surfaces, the original cross will remain where it was positioned when accepted, and a new cross is created which will follow the light pen. By repeating this process as often as necessary, the boundaries of all cavities may be located. The program is so designed that boundaries must be identified in a left to right sequence, and there must be an even number of boundaries on each surface. When all boundaries are located, touching the "END" text with the light pen terminates this operation. A section with the shear boundaries marked is shown in Figure I-6. Subroutines GETPTS and FORCE handle the placing of the crosses and finding the points on the die surfaces closest to the points indicated by the operator.

After all points have been entered, subroutines ADDPTS and SHRPTS are used to add two additional points on the shear surface between each set of cavity boundaries. Subroutines MERGE, FILTER, and DEFREL are then used to generate the arrays defining the deformation elements which will be used for stress analysis. When the definition of the deformation element arrays is complete, subroutine STRESS is started. This first asks the designer for the material flow stress value to be used. A default value is provided which can be used if desired. STRESS then calculates the stress on each successive deformation element, resulting in the display of the stress distribution curve and the printing of the stress results. The values printed include the coordinates of the neutral surface and center of load, the maximum and average stress, and the total vertical load.

Figure I-7 shows the material flow surfaces superimposed on the stress distribution curve. The stress distribution curve is always plotted as large as possible on the CRT, so scaling from the vertical axis is not possible.

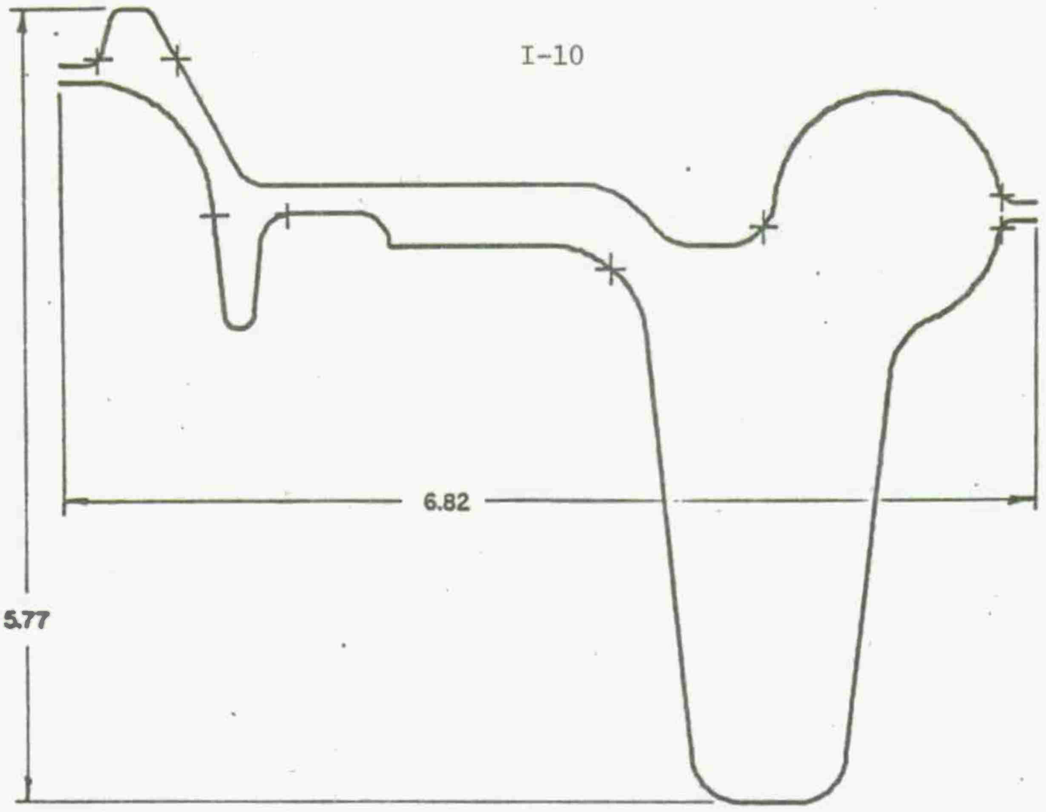
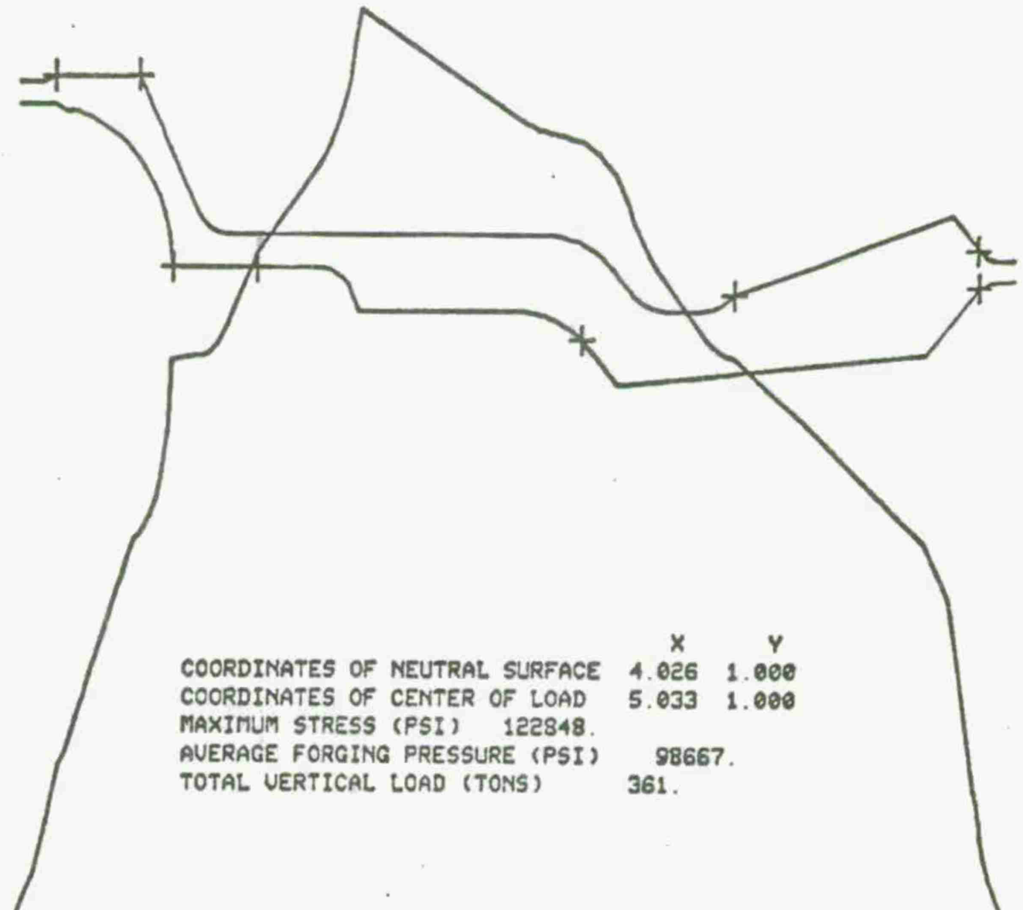


FIGURE I-6. CRT DISPLAY OF TYPICAL SECTION. SHEAR BOUNDARIES ARE MARKED WITH CROSSES. (Dimensions shown were added by hand)



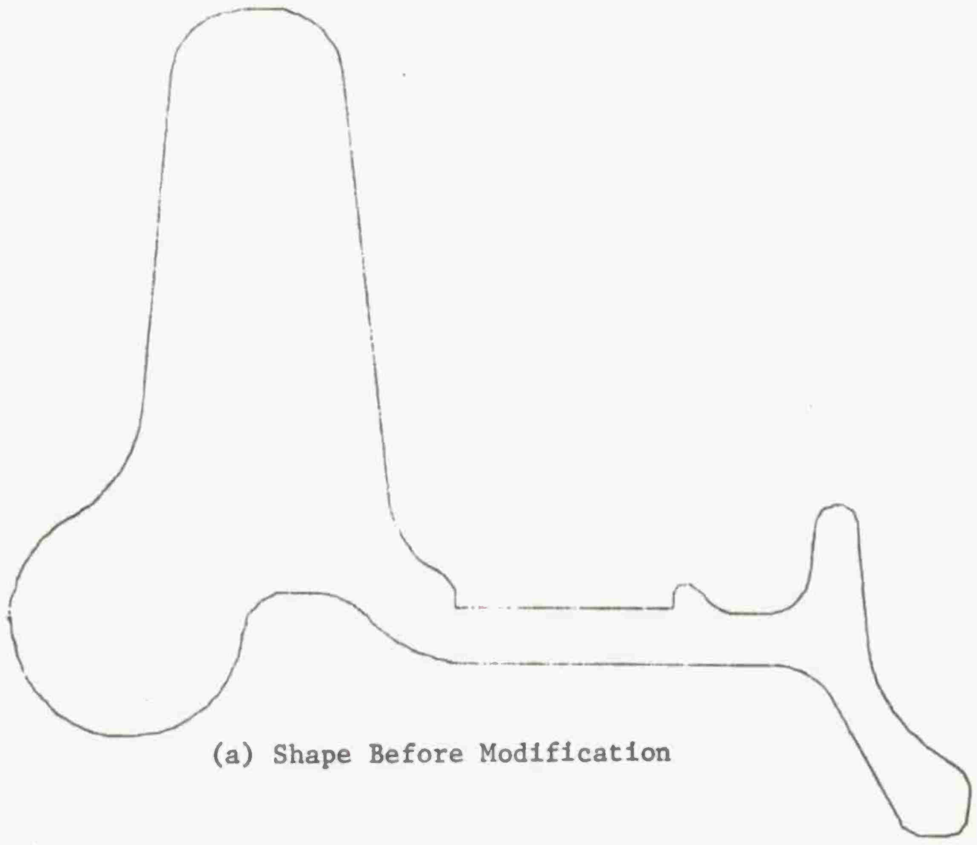
```
                                      X       Y
COORDINATES OF NEUTRAL SURFACE  4.026   1.000
COORDINATES OF CENTER OF LOAD   5.033   1.000
MAXIMUM STRESS (PSI)   122848.
AVERAGE FORGING PRESSURE (PSI)    98667.
TOTAL VERTICAL LOAD (TONS)      361.
```

FIGURE I-7. SHEAR SURFACES SUPERIMPOSED ON STRESS DISTRIBUTION DIAGRAM

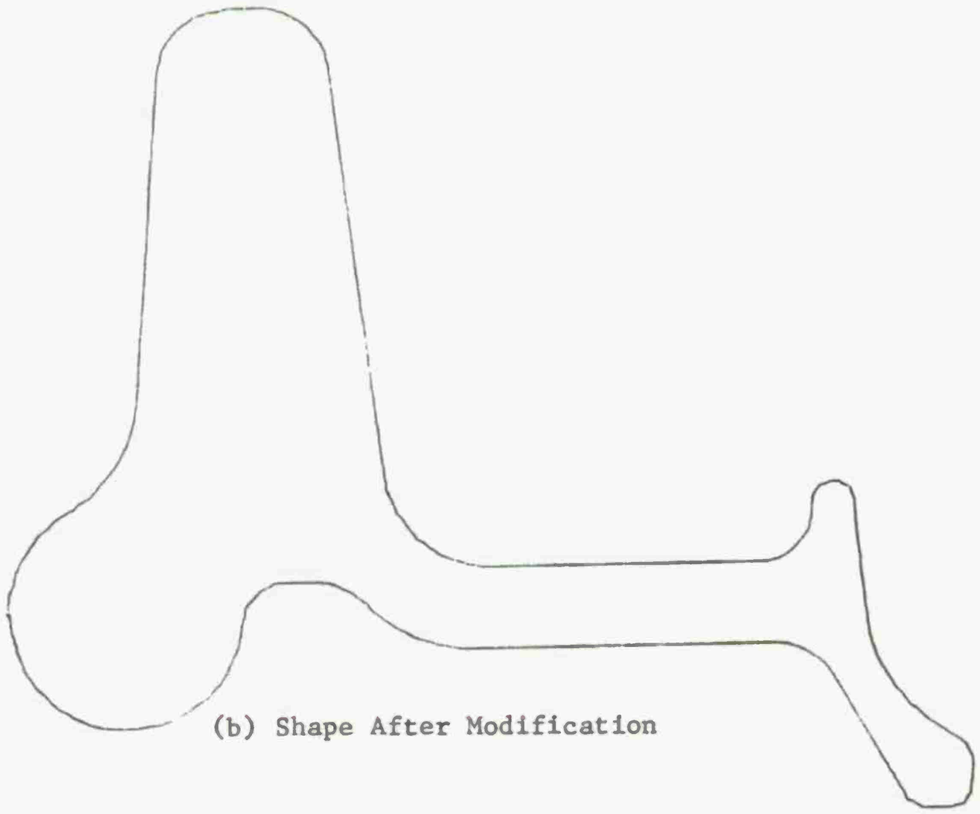
When all stress and load values are computed and displayed, the operator is asked if he wishes to save the finished die profile. If he responds with a "Y" (yes), the upper and lower profiles for the section are copied to the NC preparation file. The results for the load and center of load for the section are added to sums maintained for the total load on the entire part. The program then loops and requests the operator to indicate what function is desired next.

If option 4, Preform, is selected, subroutine PFPREP is called. This allows the designer to modify the polygon by adding and/or deleting points. Points can be added at the intersection of two lines, or as fillet or corner points anywhere along the polygon surface. A new rib can be added opposite an existing rib. Any point specified can also be deleted. Figure I-8 shows typical before and after views of a section modified by PFPREP. When the basic polygon modification is completed, subroutine PREFRM is used to alter the relationship and magnitude of the polygon coordinates. PREFRM does not permit points to be added or deleted, but it does allow them to be moved relative to one another. Sub-entities of the polygon, such as ribs, webs, or radii, can be modified as a group. Among the features of PREFRM is an "automatic" mode which attempts to design the preform without guidance from the operator. The preform shape shown in Figure I-10 was made using this automatic mode. The preform shown in Figure I-9 was made using the automatic mode followed by further designer-directed modifications. Other features of PREFRM include (1) the ability to make hard-copy plots of the CRT display, (2) allow the operator to move the preform relative to the finish die with the light pen to evaluate how well the preform nests with the finish die, and (3) allow the operator to save the completed preform polygon design on a disk file.

When PREFRM is completed, control returns to TRACKS. TRACKS then lists the options available and awaits for the designer to enter his choice. When Option 5, Done, is selected, TRACKS outputs the total load statistics for all sections analyzed, calculates the energy required for the forging, and the returns to the "CAD, CAM, DONE" inquiry.



(a) Shape Before Modification



(b) Shape After Modification

FIGURE I-8. MODIFICATION OF PREFORM POLYGON USING INTERSECT AND DELETE FUNCTIONS OF SUBROUTINE PFPREP

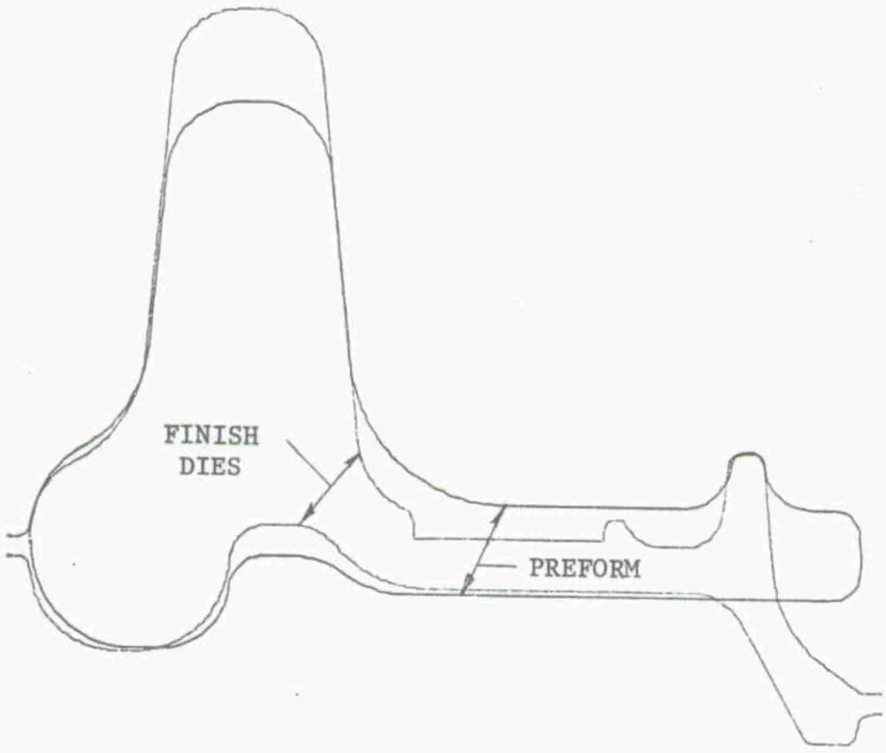


FIGURE I-9. EXAMPLE OF PREFORM DESIGNED USING SUBROUTINE PREFRM

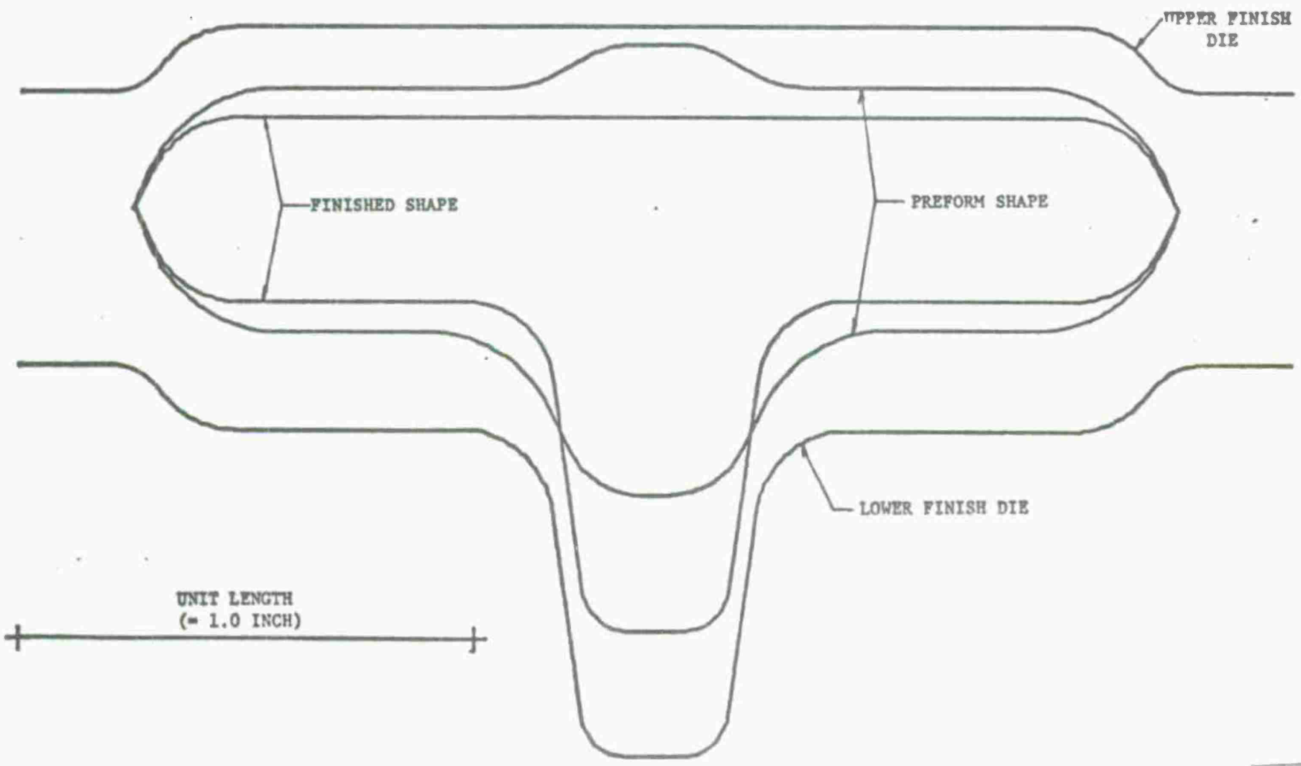


FIGURE I-10. PREFORM AND FINISHED PART SHAPES SUPERIMPOSED ON FINISHING DIES

The following summarizes the capabilities of the preform design system. Additional details are given in Appendix VII for subroutine PFPREP and PREFRM.

I. Add or delete points on polygon.

   A. Intersect - Generate a new point at the intersection of two lines defined by four points.

   B. Delete - Eliminate the point(s) indicated.

   C. Add Point - Add a new point where indicated on the part surface.

   D. Add Rib - Add four new points defining a rib. The new rib will be located opposite the rib indicated.

II. Modify relationship of points.

   A. Automatic - Expand all radii, shrink all ribs, and balance volume.

   B. Values - Print X, Z, and R values for points indicated.

   C. Balance - Adjust webs to provide volumetric balance between preform and finish sections.

   D. Ribs - Modify rib indicated.

   E. Webs - Thicken or thin web between points indicated.

   F. Radii - Expand or contract radius or radii indicated.

   G. Ends - Shift parting-line points as specified.

   H. Points - Shift points indicated by amounts specified in X and Z.

   I. Move - Allow the user to dynamically move the preform outline and/or the finish-die profiles.

   J. Save - Copy the modified section polygon to the blocker file.

   K. Plot - Copy the section and/or the finish die profiles on the X-Y recorder.

   L. Done - Terminate design and modification of current section. Remain in analysis and design phase and enter next section number if desired.

A general overview of the CAD phase of TRACK is given in Figure I-11. This indicates the major options available to the user.

The CAM (Computer-Aided Manufacturing)phase of processing is used to generate the data representing the preform model of the part. The data used is generated during the second pass through the CAD phase. (See Figure I-4). In this operation, the coordinates representing the interpolated upper and lower surfaces for each section are saved on the NC data file.

The system user dialog for the CAM phase is listed in Figure I-12. The user's responses are shown underlined. Where no user response is shown, the default value was used. Several points should be noted concerning the way in which the sections are put in sequential order. These are:

(1) In response to "Y POSITION?", the value zero, or no response indicates to the system that the default value is to be used. Thus, for the first section which was to have a Y position of 0.0, a very small but non-zero value was entered.

(2) Each transition between sections is at the same position along the Y-axis. That is, Sections 2 and 3 are both at Y = 0.375, Sections 3 and 4 are both at Y = 1.75, etc. A true vertical move is prevented at these transitions by later specifying the draft to be 7 degrees. This results in sloping the transition the amount specified.

(3) The sequencing and positioning of the sections is terminated by entering -999 in response to "SECTION NUMBER?".

(4) Because the part is symmetric about the X-axis, no flash was specified for the first end. The opposite hand section could be produced on the NC milling machine by using the mirror-image capability of the NC controller.

No graphics are used in CAM processing. The computations required to generate the surface profile are extremely lengthy, and the system becomes compute bound. For this reason, periodic messages are output to the user to assure him that the system is operating correctly. To view the results of the CAM processing, stand-alone programs NCDATA or NCPLOT may be used. These are described in Appendix IX.

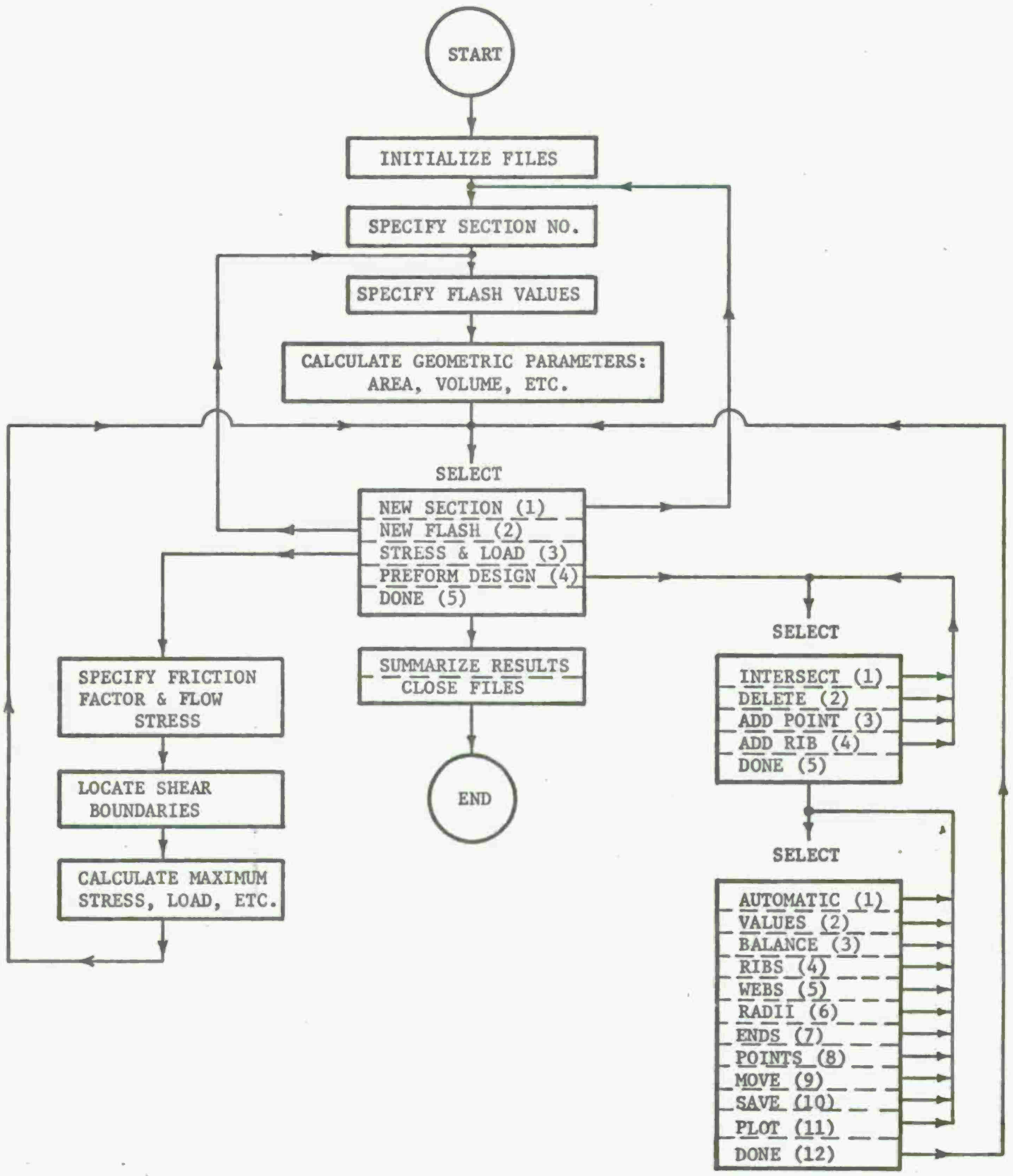


FIGURE I-11. GENERAL OPERATION OF CAD PHASE OF TRACKS SYSTEM

```
.RUN TRACKS

CAD(1), CAM(2), DONE(3)? 2


FLASH PARAMETERS

    WIDTH (0.250) ?

    THICKNESS (0.125) ?

    RADIUS (0.125) ?

    FACTOR (4.0) ?

INPUT FILE? *T130NC.DAT

UPPER OF LOWER SURFACES? (U/L) U

SECTION NUMBER? (    1) 2

Y POSITION? (-100.000) .0001

FLASH ON THIS END? (Y/N) N

SECTION NUMBER? (    2)

Y POSITION? (   0.000) .375

SECTION NUMBER? (    2) 3

Y POSITION? (   0.375)

SECTION NUMBER? (    3)

Y POSITION? (   0.375) 1.75

SECTION NUMBER? (    3) 4

Y POSITION? (   1.750)

SECTION NUMBER? (    4)

Y POSITION? (   1.750) 3.5

SECTION NUMBER? (    4) 5

Y POSITION? (   3.500)

SECTION NUMBER? (    5)

Y POSITION? (   3.500) 4.75

SECTION NUMBER? (    5) 6
```

FIGURE I-12. SYSTEM-USER DIALOG FOR GENERATION OF NC CUTTER PATH DATA FOR PREFORM MODEL

```
Y POSITION? (    4.750)

SECTION NUMBER? (     6)

Y POSITION? (    4.750) 7.5

SECTION NUMBER? (     6) -999

FLASH ON THIS END? (Y/N) Y

OUTPUT FILE FOR NC TAPE? *TEST.NC1

RESOLUTION (0.010)?

INITIAL Z-DISTANCE FROM PART SURFACE? 2

DRAFT ANGLE? 7

RADIUS OF BALL END MILL? .25

X-Z PROCESSING COMPLETED

PATH  10 COMPLETED

PATH  20 COMPLETED

PATH  30 COMPLETED

PATH  40 COMPLETED

PATH  50 COMPLETED

PATH  60 COMPLETED

PATH  70 COMPLETED

PATH  80 COMPLETED

PATH  90 COMPLETED

PATH 100 COMPLETED

CAD(1), CAM(2), DONE(3)? 3
```

FIGURE I-12. (Continued)

APPENDIX II

DESCRIPTIONS OF GENERAL-PURPOSE COMPUTER PROGRAMS

APPENDIX II

DESCRIPTIONS OF GENERAL-PURPOSE COMPUTER PROGRAMS(*)

The geometrical properties of forging cross sections must be known in order to determine the volume of a given forging. When the sections of forgings are viewed closely, it is found that they have cross-sectional areas bounded by a series of straight-lines and arcs of circles (fillet or corner). In order to calculate the surface area, or the perimeter of given forging cross sections, it is necessary to develop hardware-independent subroutine programs which can handle any cross-sectional geometry. Ideally, these subroutines must generate coordinate data so that they can be interphased to any automatic drafting system by means of customized postprocessor routines. Keeping the above requirements in mind, and taking a modular viewpoint, several subroutines were written to calculate area, center of gravity, perimeter of polygons bounded by straight-lines and circular arcs, and to plot these points to any specified scale factor. Brief descriptions of these routines are included in this appendix.

Calculation of the Area of a Cross Section

The area of any polygon, as shown in Figure II-1, may be obtained by the formula:

$$A_S = \frac{1}{2} \quad (x_2y_1 - x_1y_2) + (x_3y_2 - x_2y_3) + \ldots + (x_ny_{n-1} - x_{n-1}y_n) + (x_1y_n - x_ny_1) \qquad \text{(II-1)}$$

where x_1, x_2, ..., x_n and y_1, y_2, ..., y_n are coordinates of consecutive corners of the polygon with respect to a Cartesian coordinate system. A convenient

(*) A portion of this Appendix was originally prepared for the Air Force Program on "Manufacturing Methods for a Computerized Forging Process for High-Strength Materials", AFML-TR-73-284, prepared by Battelle's Columbus Laboratories under Contract No. F33615-71-C-1689, January, 1974. It is included here for the sake of completeness.

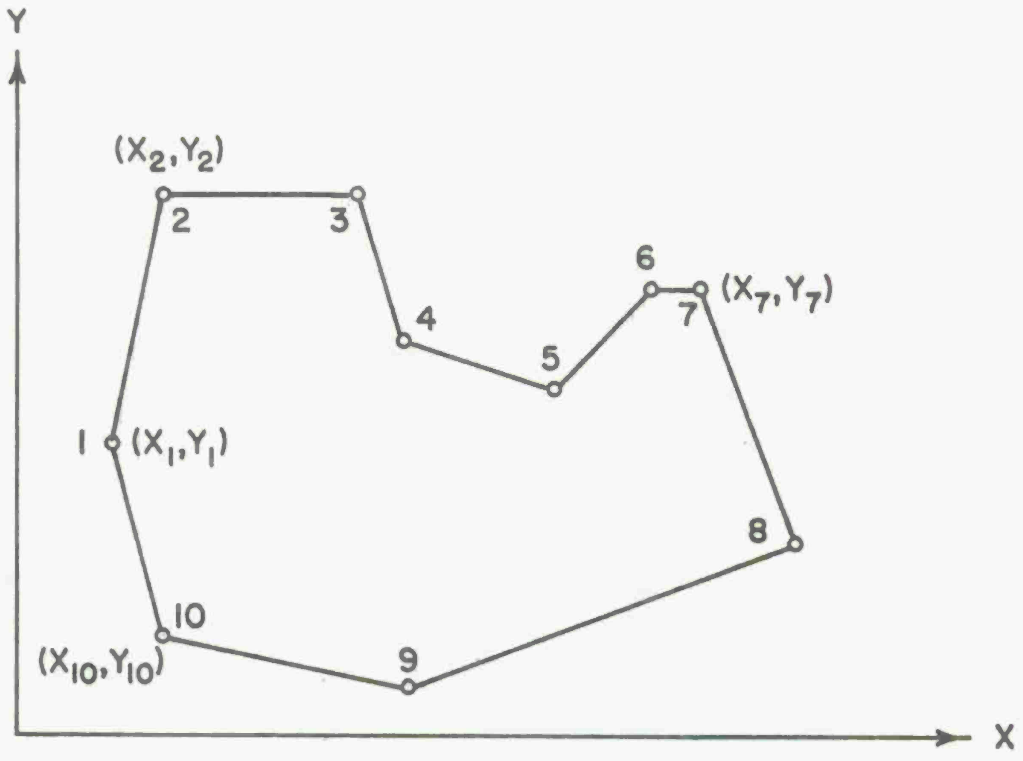


FIGURE II-1. DIAGRAM OF A POLYGON AND A RECTANGULAR COORDINATE SYSTEM DEFINING ITS CORNERS

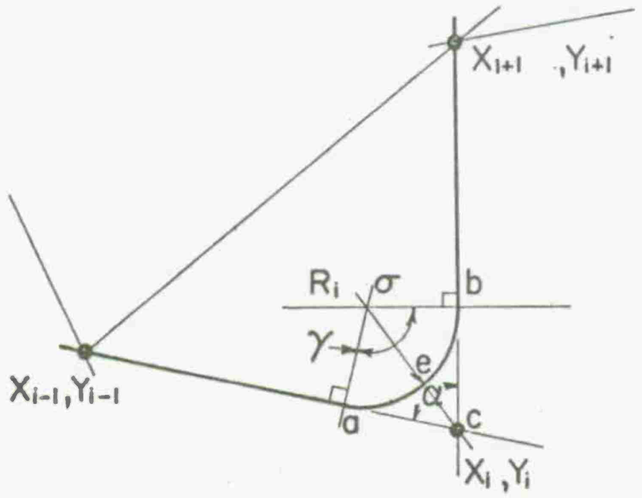


FIGURE II-2. SCHEMATIC DIAGRAM OF CORNER (i), ILLUSTRATING THE ADDITIONAL CROSS-SECTIONAL AREA DUE TO THE FILLET OF RADIUS R_i

choice of coordinate system in a nonsymmetrical part, as is the case with most aircraft structural parts, may originate at the parting line or at the datum line, and one side of the piece.

The area bounded by two straight lines and an arc of a circle, area abc as shown in Figure II-2, may be calculated by:

$$A_R = R^2 \left(\tan \frac{\gamma}{2} - \frac{\gamma}{2}\right) \quad , \qquad \text{(II-2)}$$

where R is the radius of the arc and γ is the included angle between two radii as defined in Figure II-2. If A_R is due to a fillet, then it is to be added onto the area of the polygon, A_S; on the other hand, if A_R is due to a convex corner, it should be subtracted from A_S.

Center of Gravity of a Cross Section

Center of gravity of the cross sections are used in the computation of the shape-complexity factor. The center of gravity of any polygon, as shown in Figure II-1, with respect to the y-axis is determined from the equation:

$$CG_S = \frac{\frac{1}{4}\left[\left(x_2^2y_1 - x_1^2y_2\right) + \left(x_3^2y_2 - x_2^2y_2\right) + \ldots + \left(x_n^2y_{n-1} - x_{n-1}^2y_n\right) + \left(x_1^2y_n - x_n^2y_1\right)\right]}{A_S} \qquad \text{(II-3)}$$

where $x_1, x_2, \ldots, x_n$ and $y_1, y_2, \ldots, y_n$ are the coordinates of the corners of the polygon, and A_S is calculated by Equation (II-1).

The center of gravity of an area, aebd as shown in Figure II-3, bounded by two straight lines and the arc of a circle may be calculated by:

$$\Delta x_i = R_g \cos \theta \qquad \text{(II-4)}$$

where R_g is the radius to the center of gravity from the corner and θ is the angle between the direction of R_g and the x-axis, as shown in Figure II-3. Thus,

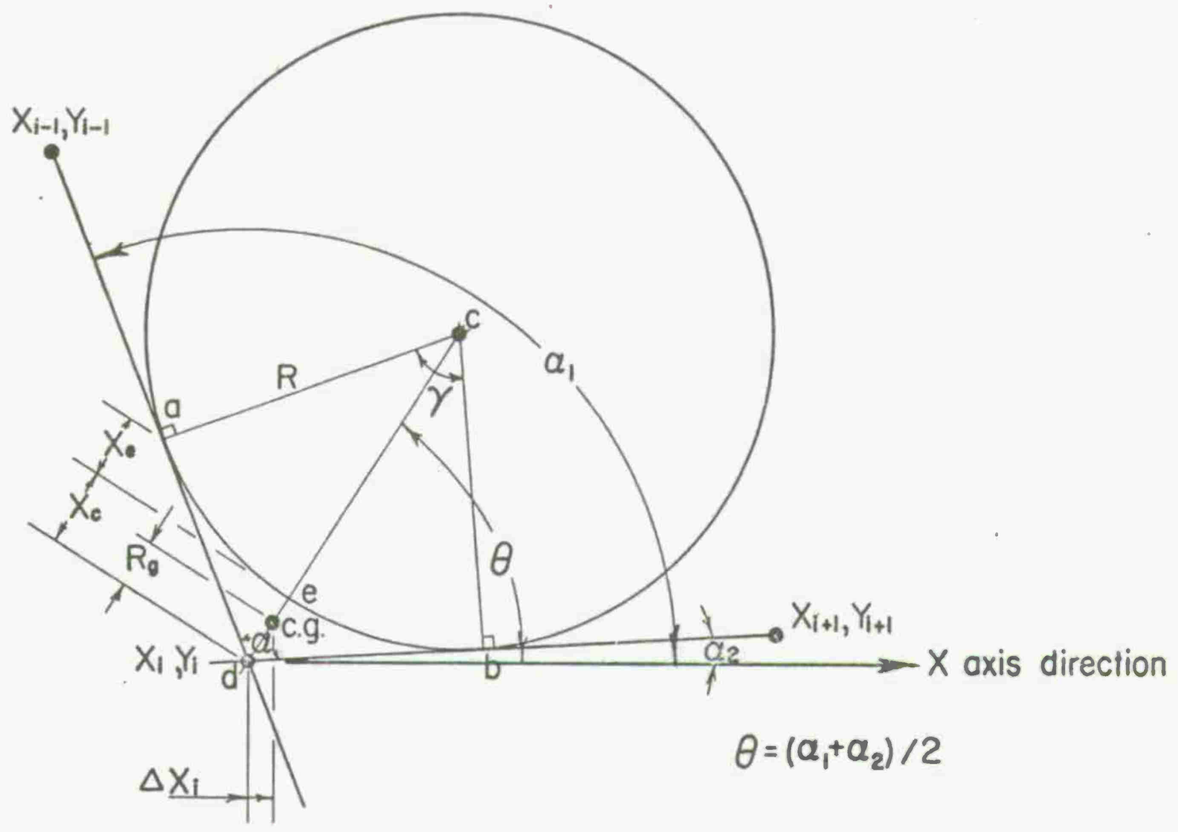


FIGURE II-3. SCHEMATIC DIAGRAM SHOWING THE LOCATION OF THE CENTER OF GRAVITY OF A FILLET AREA

when Δx_i is added into x_i, the location of the center of gravity of the area, aebd, is defined. Since

$$x_e = R\left(1 - \cos\frac{\gamma}{2}\right), \text{ and } x_c = R\left(\frac{1}{\cos\frac{\gamma}{2}}\right)$$

as defined in Figure II-3, then

$$R_g = \frac{\frac{2}{3}x_c \cdot A_T + (x_c + x_R)\,A_{RR}}{A_R} \tag{II-5}$$

where

A_R is obtained from Equation (II-2)

$$A_T = x_c^2 \tan\left(\frac{\pi-\gamma}{2}\right) \tag{II-6a}$$

$$A_{RR} = A_R - A_T \tag{II-6b}$$

and

$$x_R = \frac{\frac{x_e^3}{3}\tan\frac{\alpha}{2} + \frac{x_c x_e^2}{2}\tan\frac{\alpha}{2} + \frac{(2Rx_e - x_e^2)^{3/2}}{3} - \frac{R}{2}\left[(x_e - R)\sqrt{2Rx_e - x_e^2} + \frac{R^2\gamma}{2}\right]}{A_{RR}/2} \; . \tag{II-7}$$

The area, A_{RR}, given by Equation (II-6b) may also be expressed as:

$$A_{RR} = x_e^2 \tan\frac{\alpha}{2} + 2x_e x_c - (x_e - R)\sqrt{2Rx_e - x_e^2} - R^2\frac{\gamma}{2} \quad . \tag{II-8}$$

The angle, α, as defined in Figure II-3 is:

$$\alpha = \pi - \gamma \quad .$$

Perimeter of a Cross Section

The perimeter of any polygon, the coordinates of whose corners are $x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n$, may be calculated by:

$$P_S = \left[\sum_{i=2}^{n}\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}\right] + \sqrt{(x_n - x_1)^2 + (y_n - y_1)^2} \quad . \tag{II-9}$$

At each corner, this value must be reduced by:

$$R\gamma - 2R \tan \frac{\gamma}{2} \quad . \tag{II-10}$$

This is due to the radius at that particular corner; therefore,

$$P = P_S + \sum_{i=1}^{n} |R_i| \left(\gamma - 2 \tan \frac{\gamma}{2}\right) \quad . \tag{II-11}$$

Equation (II-11) will give the correct perimeter of a cross section, i.e., of a planar shape bounded by a succession of straight-lines and arcs of circles.

Fitting Circular Arcs to a Polygon

When working with planes defined as polygons with radii associated with each corner or fillet, there are a number of places where it is necessary to find the center of a radius, the tangency points of the arc with the polygon, or the subtended angle of the arc. Such instances occur in FITARC or DECUSP, for example. The following summarizes the mathematical derivations used in the programming of CENTER which finds the center and tangency points. All the symbols used are defined in Figure II-4, and this figure will be referred to implicitly throughout the following discussion.

Knowing the coordinates of the three points which define a corner or fillet of a polygon, the angles of each side can be found as:

$$\theta_B = \tan^{-1} [(Y_{i-1} - Y_i)/(X_{i-1} - X_i)] \tag{II-12}$$

$$\theta_E = \tan^{-1} [(Y_{i+1} - Y_i)/(X_{i+1} - X_i)] \tag{II-13}$$

The angle of the bisector can then be found as the average of θ_B and θ_E or

$$\theta_A = (\theta_B + \theta_E)/2 \quad . \tag{II-14}$$

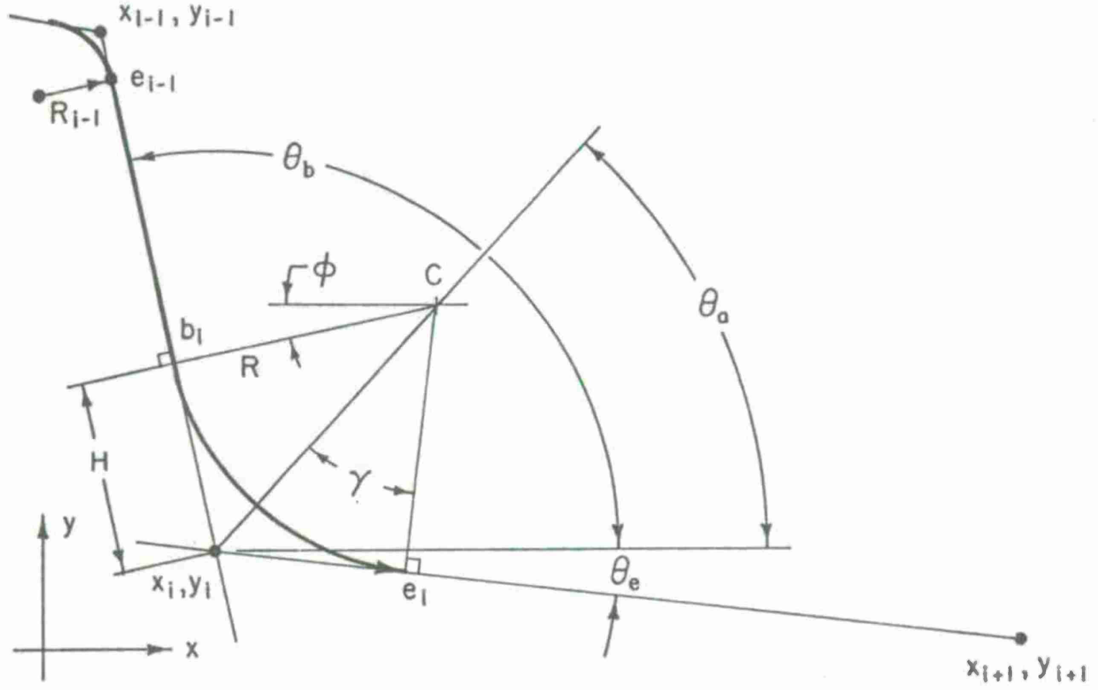


FIGURE II-4. FITTING AN ARC TO A POINT ON A POLYGON

Since the center of the arc lies along the bisector and the tangency points intersect the sides of the polygon at right angles, half of the angle subtended by the arc can be found as follows:

Since $\pi = (\theta_A - \theta_E) + \pi/2 + \gamma$, (II-15)

$$\gamma = \theta_E - \theta_A + \pi/2 \quad . \qquad \text{(II-16)}$$

θ_E is subtracted in the above expression since the value returned by the function is a signed quantity indicating both magnitude and direction.

Using expression for θ_A in the above gives:

$$\gamma = \theta_E - (\theta_B + \theta_E)/2 + \pi/2 \qquad \text{(II-17)}$$

$$\gamma = (\theta_E - \theta_B + \pi)/2 \quad . \qquad \text{(II-18)}$$

Since the total included angle approaches π as $(\theta_B - \theta_E)$ approaches π, the following adjustment is necessary:

$$\text{If } \gamma > \pi/2,\ \gamma = (\theta_E - \theta_B - \pi)/2 \quad . \qquad \text{(II-19)}$$

The absolute value of γ is the magnitude of the half-subtended angle and the sign of γ indicates the direction of rotation of the arc. If $\gamma > \phi$, the arc is counterclockwise. If $\gamma < \phi$, the arc is clockwise.

The center of the radius is next found as:

$$X_c = X_i + R\cos(\theta_A)/\cos\gamma \qquad \text{(II-20)}$$
$$Y_c = Y_i + R\sin(\theta_A)/\cos\gamma \quad . \qquad \text{(II-21)}$$

The distance, H, from the center to the tangency points can be expressed as:

$$H/R = \tan\gamma \text{ or } H = R\tan\gamma \quad , \qquad \text{(II-22)}$$

using the absolute value of γ.

The tangency points themselves are then found as:

$$X_B = X_C + H \cos \theta_B \qquad \text{(II-23)}$$

$$Y_B = Y_C + H \sin \theta_B \qquad \text{(II-24)}$$

and

$$X_E = X_C + H \cos \theta_E \qquad \text{(II-25)}$$

$$Y_E = Y_C + H \sin \theta_E \qquad \text{(II-26)}$$

For the purpose of graphically displaying an arc, it is necessary to simulate the arc as a series of short, linear segments. The smaller each of these segments are, the better the representation will be. The technique for calculating the end points of the individual segments is as follows:

With r = resolution desired in thousandths of an inch, the number of segments needed is the arc length divided by the resolution, or

$$n = \text{integer part of } [1000\ R\gamma/r + .5] \quad . \qquad \text{(II-27)}$$

The angular increments can then be found as the total subtended angle divided by the number of increments, or

$$\delta = \frac{\gamma}{n} \quad . \qquad \text{(II-28)}$$

The starting angle of the arc is found by knowing the center and the first tangent point as

$$\phi = \tan^{-1} [(Y_b - Y_c)/(X_b - X_c)] \quad . \qquad \text{(II-29)}$$

Incrementing ϕ_b by the amount of δ for n times gives the coordinate points on the arc as:

$$X_{aj} = X_c + R \cos (\phi + j\delta) \qquad \text{(II-30)}$$
$$Y_{aj} = Y_c + R \sin (\phi + j\delta) \qquad \text{(II-31)}$$

where $j = 0,1,2, \ldots n$. (II-32)

Finding the Minimum Distance from a Point to a Point on a Polygonal Surface

Given a series of points A, B, C, ... N, which define a surface from left to right, a point on the surface can be found which is the closest to some other point, P_1, by the following procedure.

From each pair of adjacent points, i.e., C and D, a triangle can be constructed with P_1 as the third point (see Figure II-5). The lengths of the sides of this triangle are then determined by Equations (II-33), (II-34), and (II-35). With this information, by using the Law of Cosines, the angles α and β opposite P can be determined by Equations (II-36) and (II-37). If either of these angles is obtuse, a normal to line CD passing through P does not exist which lies between points C and D. In such a case, the point on the surface closest to P is one of the points (C or D) defining the surface. Comparing the lengths L_1 and L_2 to the previously found or established minimum will indicate whether or not a new point nearest to P has been found.

$$L_1 = ((X_{P2} - X_c)^2 + (Y_{P1} - Y_C)^2)^{1/2} \quad \text{(II-33)}$$

$$L_2 = ((X_{P1} - X_D)^2 + (Y_{P1} - Y_D)^2)^{1/2} \quad \text{(II-34)}$$

$$L_3 = ((X_D - X_C)^2 + (Y_D - Y_C)^2)^{1/2} \quad \text{(II-35)}$$

$$\cos \alpha_1 = (L_1{}^2 + L_3{}^2 - L_2{}^2)/(2 \cdot L_1 \cdot L_2) \quad \text{(II-36)}$$

$$\cos \beta_1 = (L_2{}^2 + L_3{}^2 - L_1{}^2)/(2 . L_2 . L_3) \quad \text{(II-37)}$$

If α and β are acute, a normal through P exists which passes through the line defined by the surface coordinates (see Figure II-5). Therefore, the normal from P_2 intersects line L_3 between the end points I and J defining L_3. Using the fact that if 2 lines are normal to each other, the slope of one is the negative inverse of the slope of the other, the coordinates of the point of intersection, P_3, can be found. Comparing L_n to the previously determined, or established minimum value will indicate whether a new point, closer to P than any other, has been found.

The amount of searching required to find the point on the surface closest to P can be limited by using the fact that the surface is defined from left to right. Since the solution is bounded on the left side of the point in question, by either the previous solution or the initial conditions, the range R from the point in question to the left bound can be found. The solution will lie within R distance either side of P. Initially, the solution is bounded on the left by the first point of the surface.

By definition, two points defining the location of a cavity cannot lie under the same line segment which defines part of the surface. Further, the points used to indicate cavities are entered in a left to right manner. Therefore, the solution to the surface point nearest to some other point P generates a new left-hand bound for the next trial. The left-most coordinate of the first triangle to test when the next point is entered is the right-most coordinate of the triangle which contains the point closest to the point currently being considered.

When testing the sides of adjacent triangles when a normal does not exist, it is only necessary to test the first side as a new minimum. This is because the second side of one triangle will be treated as the first side of the succeeding triangle. In Figure II-6, side L_2 of triangle CDP, becomes side 1 of triangle DEP. By only testing the first side as a new minimum, triangle DEP will be found to contain the surface point closest of P. Therefore, point E will become the left-hand bound for the solution to the next point entered. In Figure II-5, EFP_2 becomes the first triangle to test when P_2 is located.

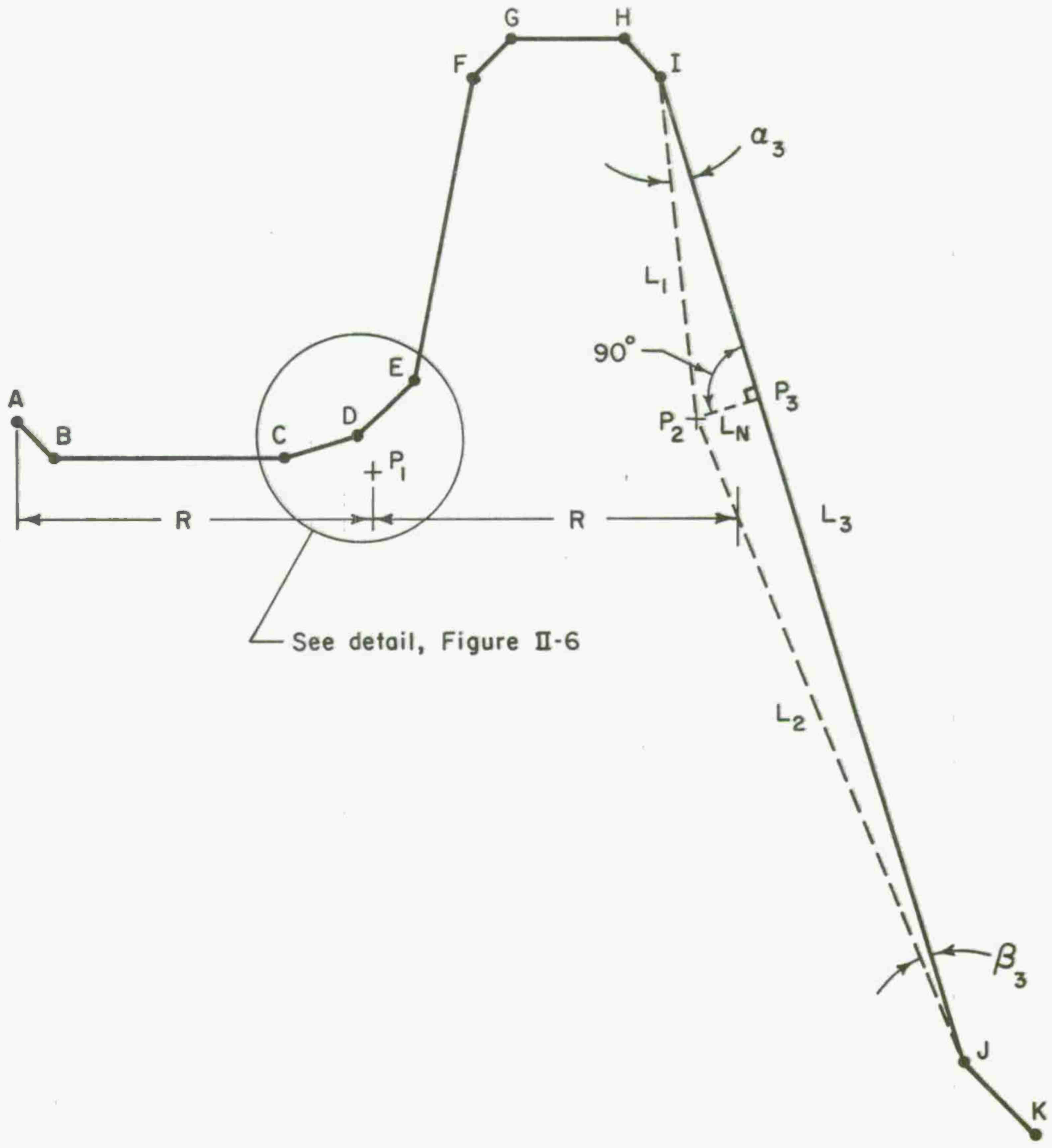


FIGURE II-5. TECHNIQUE USED TO FIND THE POINT ON A POLYGONAL SURFACE CLOSEST TO SOME INDICATED POINT

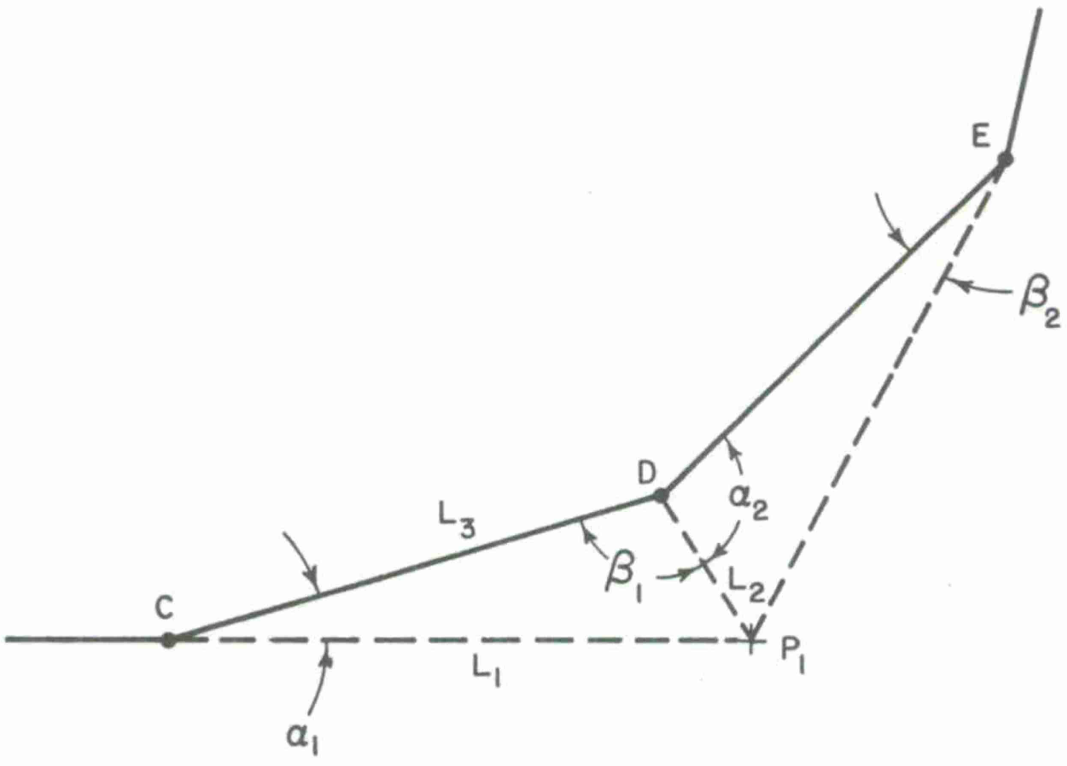


FIGURE II-6. DETAIL OF FIGURE II-5

APPENDIX III

<u>STRESSES AND LOADS IN FORGING</u>

APPENDIX III

STRESSES AND LOADS IN FORGING(*)

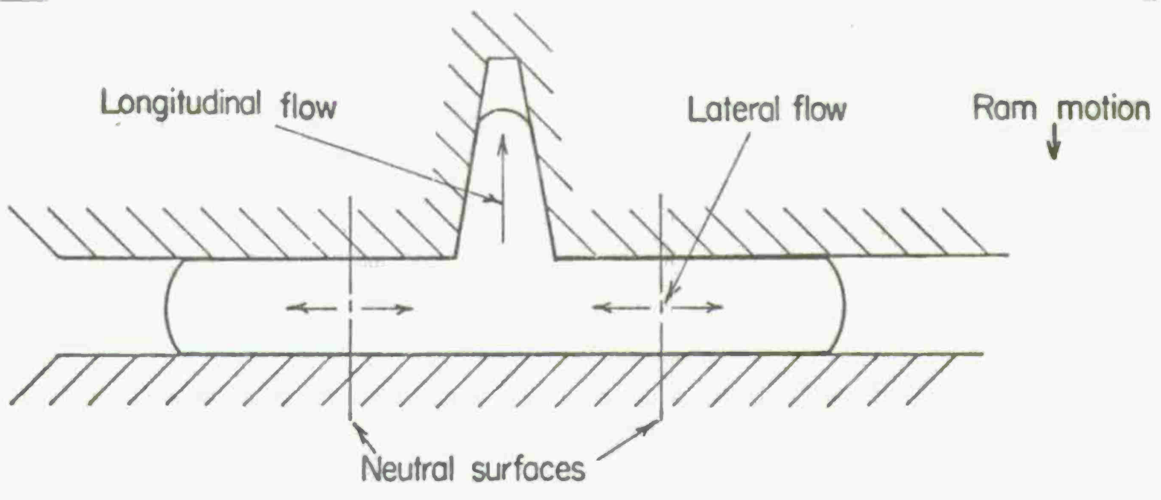
The equations given in this appendix are used in preparing the computer program subroutines for calculating stresses and loads in forging. These equations are obtained by using the Slab Method of analysis.

The approach for analyzing a nonsymmetrical structural forging is to divide the forging into several connected components called "Deformation Units". Thus, it is necessary to define the following types of metal flow.

Plane Strain, lateral flow (upsetting)
Plane Strain, longitudinal flow (extrusion)
Axisymmetric, lateral flow (upsetting)
Axisymmetric, longitudinal flow (extrusion).

Any given structural forging can be divided into units of deformation, represented by separate or combined metal-flow behaviors listed above. Lateral flow occurs, at a given location in forging, when metal flow is of upsetting type, i.e., it is predominantly perpendicular to the die motion, as shown in Figure III-1. Longitudinal flow exists when metal flow is of extrusion type, i.e., it is parallel with die motion. Since the stresses and loads are calculated at the point of maximum die closure when all cavities are assumed to be filled, extrusion type flow is not considered in the following discussion.

(*) This appendix is given here for sake of completeness and it was originally prepared for the Air Force Program on "Manufacturing Methods for a Computerized Forging Process for High-Strength Materials", AFML-TR-73-284, prepared by Battelle's Columbus Laboratories under Contract No. F33615-71-C-1689, January, 1974.



a. Example of Plane-Strain, Longitudinal Flow

FIGURE III-1. ILLUSTRATION OF LATERAL AND LONGITUDINAL METAL FLOW IN FORGING A RIB-WEB TYPE COMPONENT

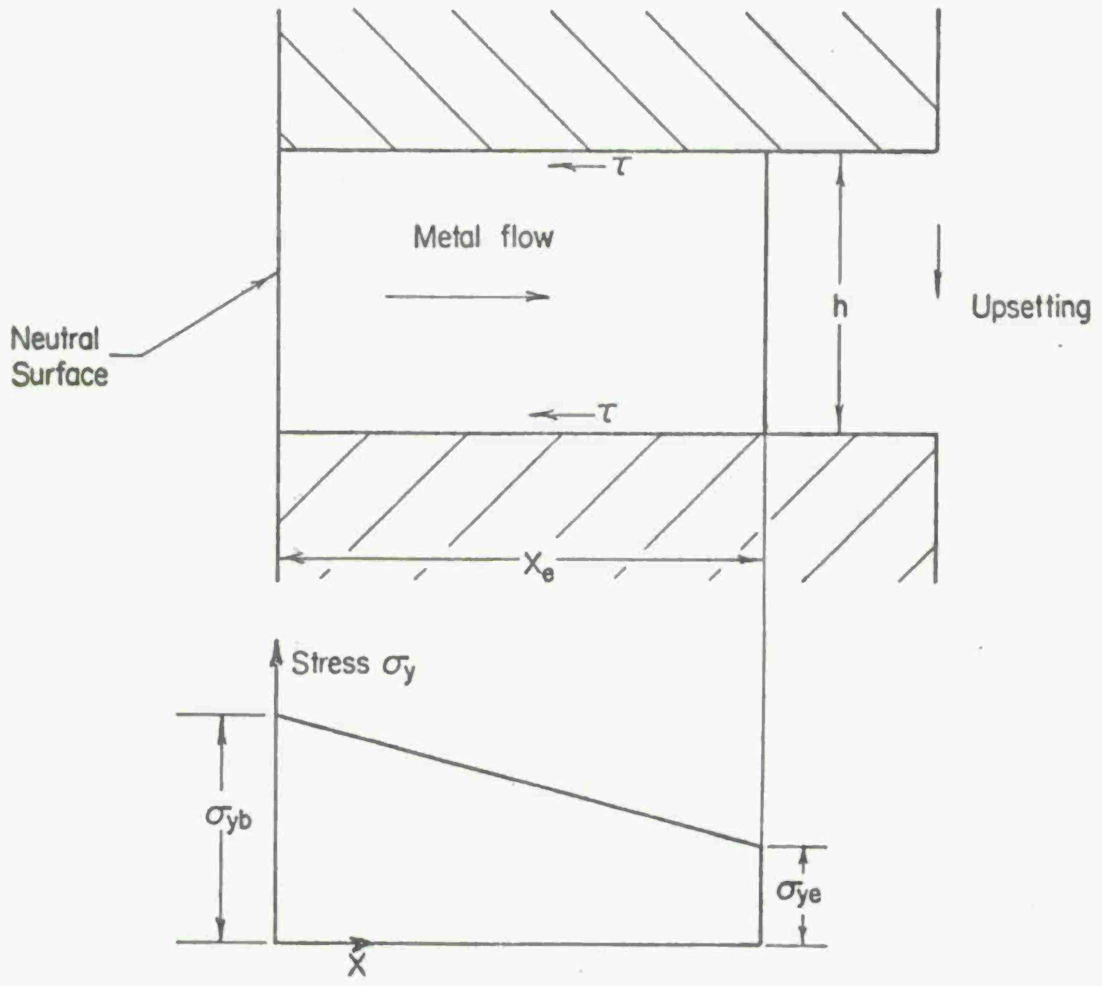


FIGURE III-2. VERTICAL FORGING STRESS DISTRIBUTION IN A DEFORMATION UNIT REPRESENTING UPSET FORGING BETWEEN HORIZONTAL DIES

Plane-Strain Deformation

Plane-strain conditions prevail when metal flow is two dimensional, for instance, in vertical and horizontal direction in Figure III-1. Here, the forging is long enough that no metal flow exists perpendicular to the plane of the paper in Figure III-1.

Lateral Flow Between Horizontal Dies (Upset Forging)

The deformation unit for plane-strain upset forging between horizontal dies is illustrated in Figure III-2. Using the symbols given in Figure III-2, the vertical forging stress, σ_y, is given by:

$$\sigma_y = \frac{2\tau}{h} (X_e - X) + \sigma_{ye} \quad , \qquad \text{(III-1)}$$

where τ = friction shear stress = $f\bar{\sigma}$

$\bar{\sigma}$ = flow stress of the forged material

f = friction factor, determined experimentally

σ_{ye} = vertical stress at the end of forging, for $X = X_e$

σ_{ye} is determined from the adjacent deformation unit

h = material thickness.

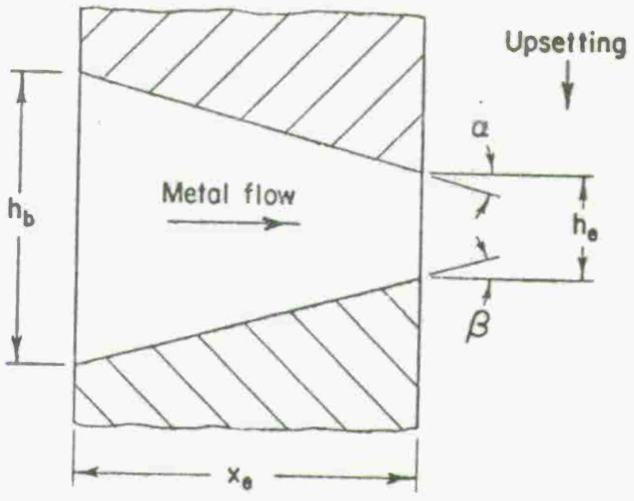
The forging load per unit depth, P, is given by:

$$P = \left(X_e \cdot \sigma_{ye}\right) + \frac{\tau X_e}{h} \quad . \qquad \text{(III-2)}$$

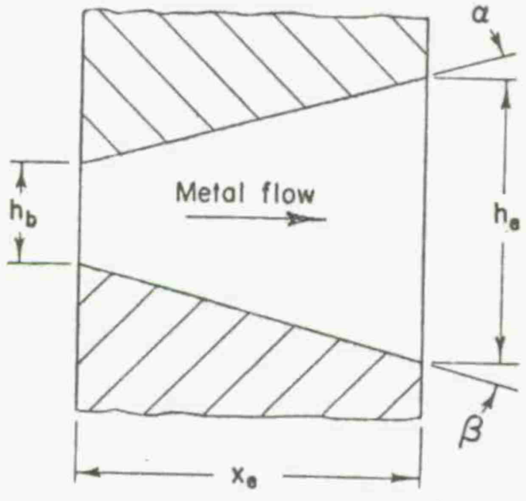
Lateral Flow Between Inclined Dies

The deformation unit for plane-strain upset forging between inclined dies is illustrated in Figure III-3. With the symbols given in Figure III-3, the vertical forging stress, σ_y, is given by:

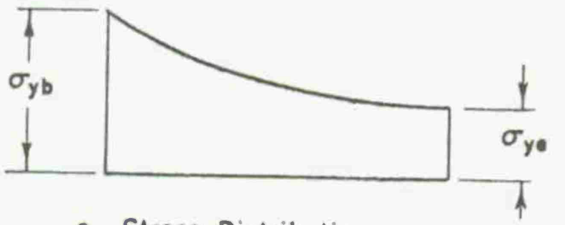
$$\sigma_y = \frac{K_2}{K_1} \ln \left(\frac{h_e}{h_b + K_1 X} \right) + \sigma_{ye} \quad , \qquad \text{(III-3)}$$



a. Converging Flow ($\alpha < o, \beta < o$)



b. Diverging Flow ($\alpha > o, \beta > o$)



c. Stress Distribution

FIGURE III-3. DEFORMATION UNITS FOR UPSET FORGING BETWEEN INCLINED PLATENS FOR PREDOMINANTELY HORIZONTAL METAL FLOW

where $K_1 = (\tan\alpha + \tan\beta)$

$$K_2 = -\frac{2}{\sqrt{3}}\bar{\sigma}K_1 + \tau\,(2 + \tan^2\alpha + \tan^2\beta).$$

The forging load per unit depth is given by:

$$P = -\frac{K_2}{K_1}\cdot\frac{1}{K_1}\left[h_e(\ln h_e - 1) - h_b(\ln h_b - 1)\right] + \left[\sigma_{ye} + \frac{K_2}{K_1}\ln h_e\right] X_e \ . \qquad \text{(III-4)}$$

Equations (III-3) and (III-4) are valid for all types of lateral flow illustrated in Figures III-3 and III-4. In each case, the appropriate signs must be used for the angles α and β, as given below:

(1)	Converging Flow (Figure III-3a):	$\alpha < 0,\ \beta < 0$
(2)	Diverging Flow (Figure III-3b):	$\alpha > 0,\ \beta > 0$
(3)	Ascending Flow (Figure III-4a):	$\alpha > 0,\ \beta < 0$
(4)	Descending Flow (Figure III-4b):	$\alpha < 0,\ \beta > 0$.

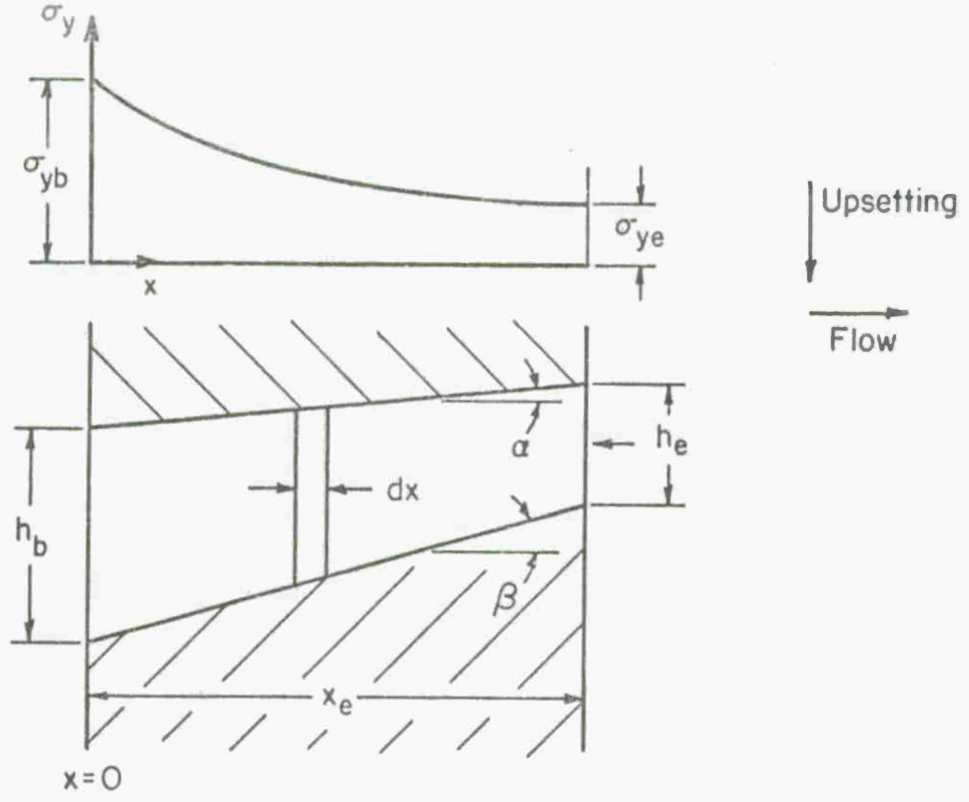
The equations, given above, for calculating load and stress in a deformation unit for upsetting have been programmed as a subroutine in FORTRAN IV.

Axisymmetric Deformation

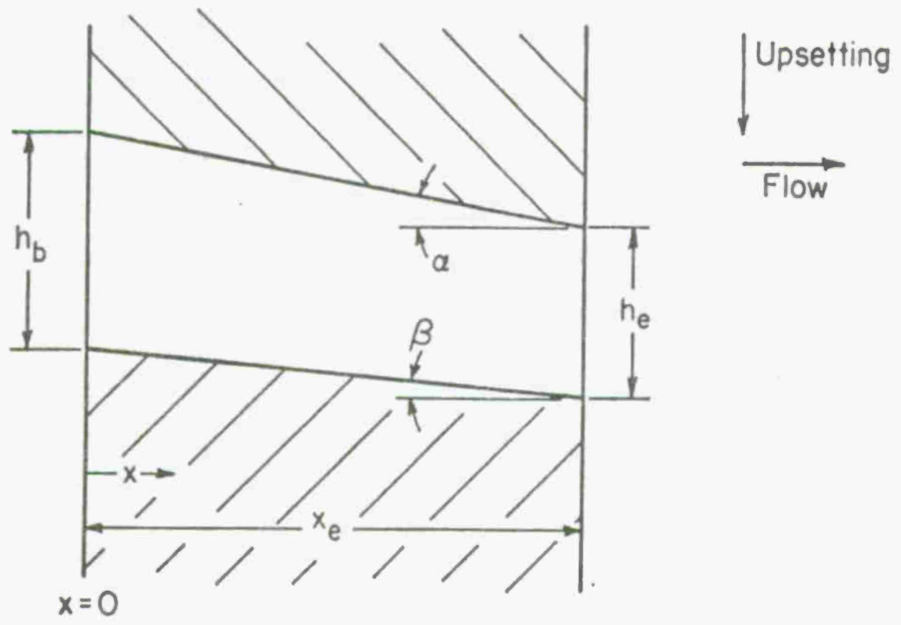
Axisymmetric conditions prevail when metal flow is in three dimensions, but symmetric with respect to the axis of a part, as it is the case in forging a round part.

Lateral Flow Between Horizontal Dies (Upset Forging

The deformation unit for axisymmetric upset forging between horizontal dies is illustrated in Figure III-5 for outward flow. In this case, a ring with internal radius r_b and external radius r_e is deformed in such a way that the external radius, r_e, increases while the internal radius, r_b, remains constant. The internal surface of the ring is the neutral surface, i.e., the surface which remains stationary during deformation.



a. Ascending Flow ($\alpha > o$, $\beta < o$)



b. Descending Flow ($\alpha < o$, $\beta > o$)

FIGURE III-4. DEFORMATION UNITS FOR UPSET FORGING BETWEEN INCLINED PLATENS FOR ASCENDING AND DESCENDING METAL FLOW

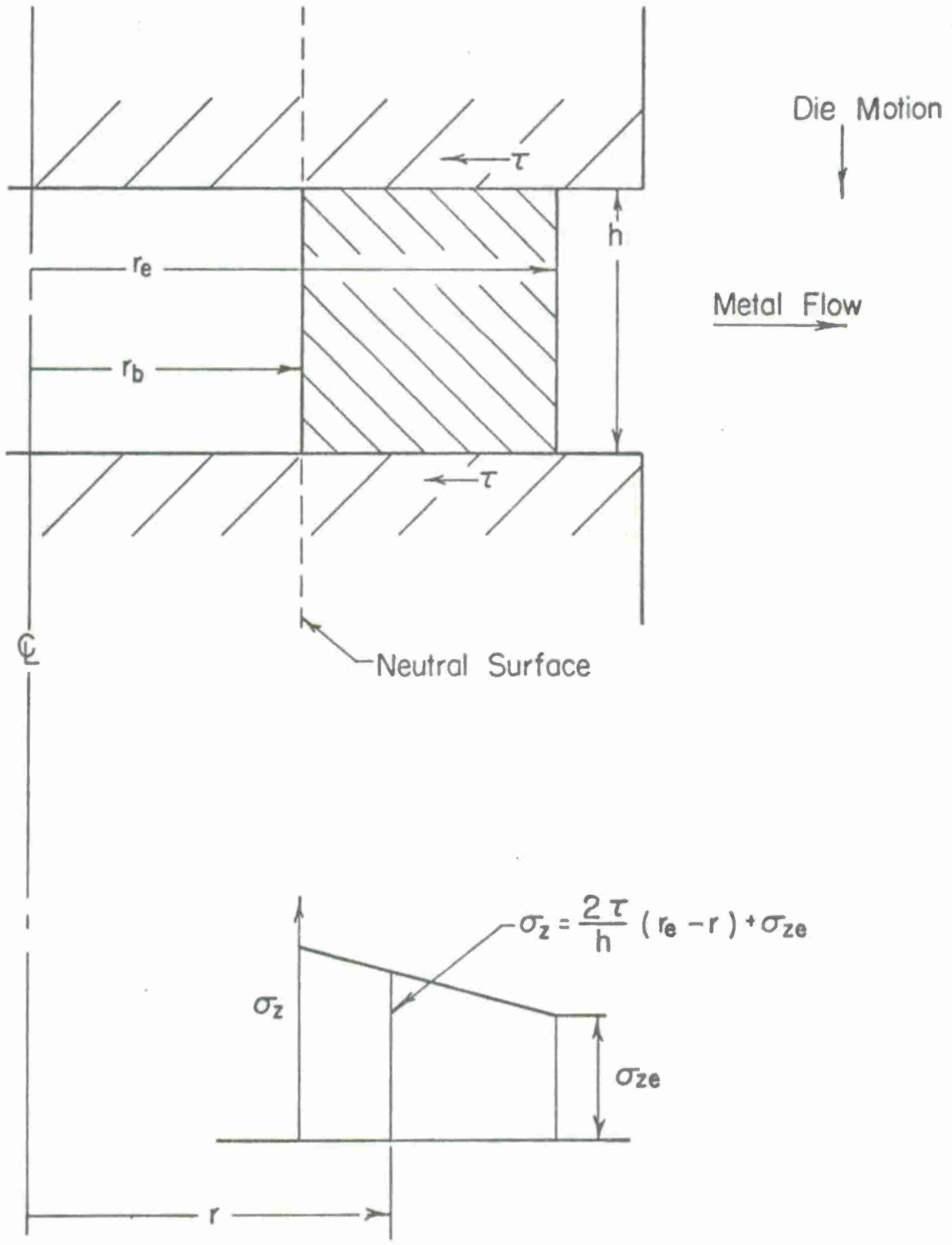


FIGURE III-5. METAL FLOW AND STRESS DISTRIBUTION IN AXISYMMETRIC LATERAL FLOW (OUTWARD) BETWEEN HORIZONTAL DIES

For outward flow, using the symbols given in Figure III-5, the axial forging stress, σ_z, is given by:

$$\sigma_z = \frac{2\tau}{h}(r_e - r) + \sigma_{ze} \quad . \qquad \text{(III-5)}$$

σ_{ze} is the axial stress at the end of the deformation unit, i.e., at $r = r_e$. σ_{ze} is determined from the adjacent deformation unit. If, at $r = r_e$, the deforming ring has a free surface, then $\sigma_{ze} = \bar{\sigma}$, flow stress of deforming material.

The forging load, P, on a ring sector of angle θ is given by:

$$P = \theta\left[\left(-\frac{2\tau}{h}\ \frac{r_e^3 - r_b^3}{3}\right) + \left(\frac{2\tau\ r_e}{h} + \sigma_{ze}\right)\left(\frac{r_e^2 - r_b^2}{2}\right)\right] . \qquad \text{(III-6)}$$

In inward flow, the neutral surface is at $r = r_e$ and r_e remains stationary while r_b is reduced as deformation proceeds. In this case, the direction of the friction shear stresses τ is reversed and the axial stress increases from the moving boundary at $r = r_b$ towards the neutral surface at $r = r_e$. The axial stress, σ_z, is given by:

$$\sigma_z = \frac{2\tau}{h}\cdot(r - r_b) + \sigma_{zb} \quad . \qquad \text{(III-7)}$$

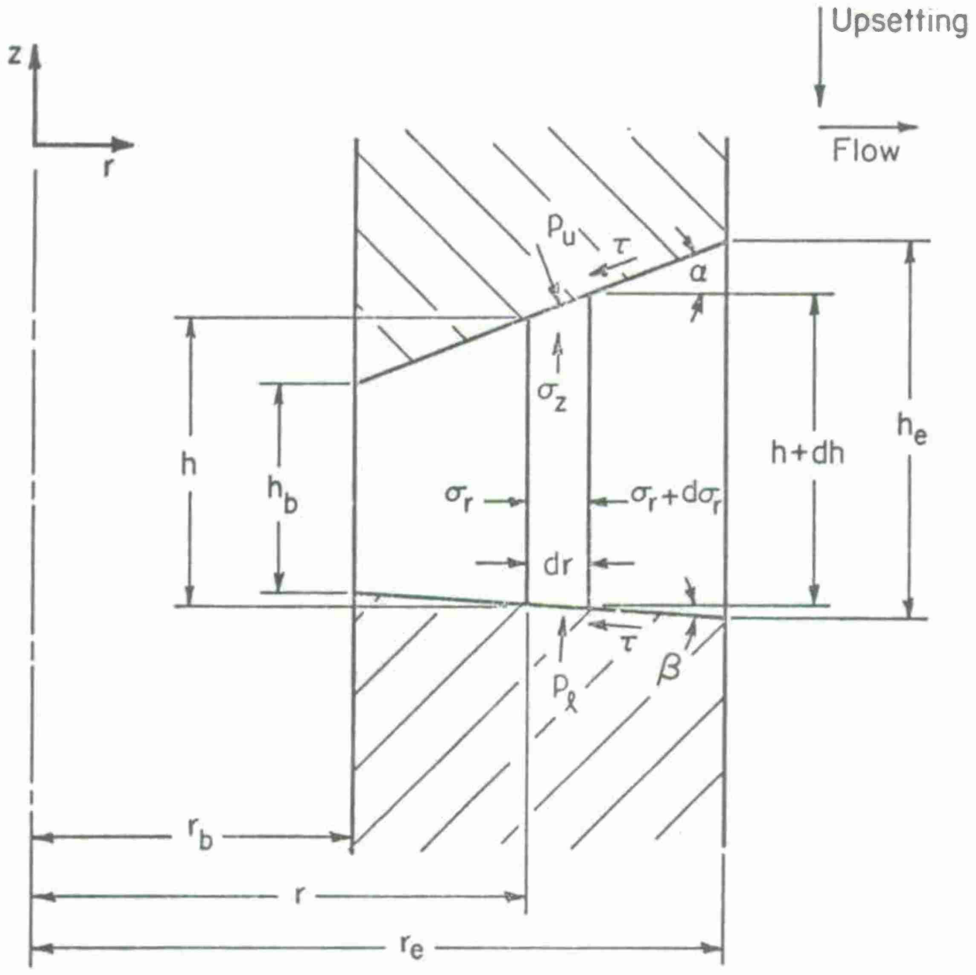
σ_{zb} is the axial stress at $r = r_b$ and it is determined from the adjacent deformation unit.

The forging load, P, is obtained by integration, and for a ring sector of angle θ it is given by:

$$P = \theta\left[\left(\frac{2\tau}{h}\cdot\frac{r_e^3 - r_b^3}{3}\right) + \left(\sigma_{zb} - \frac{2\tau\ r_b}{h}\right)\left(\frac{r_e^2 - r_b^2}{2}\right)\right] . \qquad \text{(III-8)}$$

Lateral Flow Between Inclined Dies (Upset Forging)

The deformation unit which represents axisymmetric upsetting between inclined planes is illustrated in Figure III-6 for outward flow. With the symbols used in Figure III-6, the axial stress, σ_z, is given by:



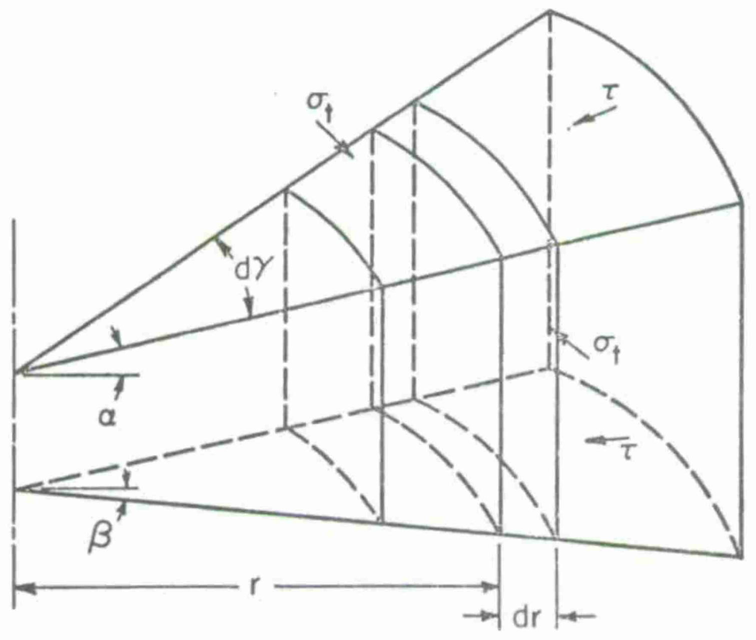


FIGURE III-6. AXISYMMETRIC UPSETTING UNDER INCLINED SURFACES (DIVERGING FLOW)

$$\sigma_z = \frac{K_2}{K_1} \ln \left(\frac{h_e}{K_3 + K_1 r} \right) + \sigma_{ze} \, , \tag{III-9}$$

where σ_{ze} = the axial stress at $r = r_e$, determined from the adjacent deformation unit

$$K_1 = \tan\alpha + \tan\beta \tag{III-10}$$

$$K_2 = -\,\bar{\sigma} \cdot K_1 + \tau \left(\frac{1}{\cos^2\alpha} + \frac{1}{\cos^2\beta} \right) \tag{III-11}$$

$$K_3 = h_b - r_b \cdot K_1 \, . \tag{III-12}$$

The upsetting load, P, over the sector of angle θ, shown in Figure III-6, is:

$$P = \int_{r_b}^{r_e} \left[\theta \frac{K_2}{K_1} r \;\; \ln \left(\frac{h_e}{K_3 + K_1 r} \right) + \sigma_{ze} \right] dr \, . \tag{III-13}$$

(Solution is most easily accomplished by numerical techniques).

As it was in case of plane-strain deformation, Equations (III-9) through (III-12) are valid for all types of lateral flow which correspond to the cases seen in Figures III-3 and III-4. In each case, the appropriate signs must be used for the angles α and β, as given below:

(1) Diverging Flow (Figure III-6): $\alpha > 0$, $\beta > 0$

(2) Converging Flow (Similar to Figure III-3a): $\alpha < 0$, $\beta < 0$

(3) Ascending Flow (Similar to Figure III-4a): $\alpha > 0$, $\beta < 0$

(4) Descending Flow (Similar to Figure III-4b): $\alpha < 0$, $\beta > 0$.

So far, only outward flow has been considered. In case of inward flow, the same Equations (III-9) through (III-13) are valid, but the sign of the friction shear stress, τ, must be reversed in all equations.

CENTER OF LOADING IN DEFORMATION UNITS

The center of loading in deformation units are useful in determining the center of loading of a complete forging. The center of loading of a deformation unit is a function of the stress distribution on the unit and the geometry of the unit. The derivations for the various units are given below.

Plane-Strain Deformation

Lateral Flow Between Horizontal Dies (Upset Forging)

The stress distribution in this case is given by Equation (III-1), which is linear in x as shown in Figure III-2. The location of the center of loading, with respect to the ordinate of Figure III-2, is given by:

$$x_{CL} = \frac{x_e \left(\sigma_{yb} + 2\sigma_{ye}\right)}{3\left(\sigma_{yb} + \sigma_{ye}\right)} \quad . \tag{III-14}$$

Lateral Flow Between Inclined Dies

The stress distribution on this deformation unit is given by Equation (III-3) which is nonlinear. Instead of attempting an analytical solution, a numerical approach is taken. As shown in Figure III-7, the deformation unit is divided into small elements and the stress at the center of each element is calculated by Equation (III-3). If there are n such elements, the center of loading is given by:

$$x_{CL} = \frac{\sum_{i=1}^{n} x_i \cdot \sigma_{yi}}{\sum_{i=1}^{n} \sigma_{yi}} \quad . \tag{III-15}$$

Axisymmetric Deformation

Lateral Flow

As seen in Figure III-8, an analytic approach to the determination of the center of loading of axisymmetric deformation units, though possible, is lengthy. Numerical methods provide an elegant and relatively simple approach. In general, a unit is divided into n elements of width Δr, as shown in Figure III-8. The stress at $r = (r_i + r_o)/2$ is calculated by the appropriate equation.

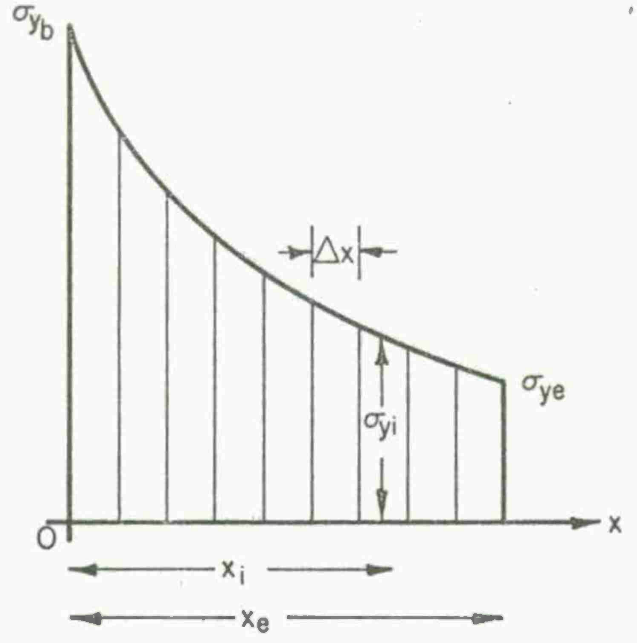


FIGURE III-7. THE DIVISION OF THE STRESS DISTRIBUTION INTO SMALL ELEMENTS IN ORDER TO CALCULATE THE CENTER OF LOADING NUMERICALLY

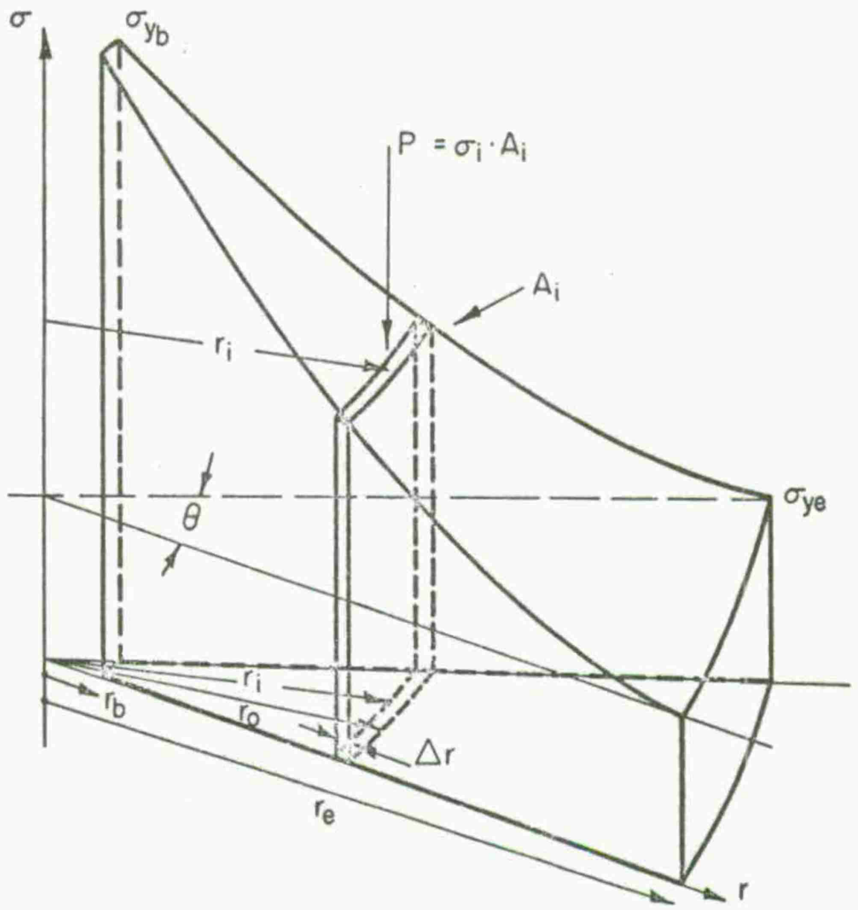


FIGURE III-8. SCHEMATIC OF LOAD ELEMENT FOR NONLINEAR AXISYMMETRIC STRESS DISTRIBUTION

The area of the element is $\frac{\theta}{2}(r_o^2 - r_i^2)$ and the center of loading is given by:

$$r_c = \frac{2 \sin(\theta/2)}{3(\theta/2)} \cdot \frac{r_o^3 - r_i^3}{r_o^2 - r_i^2} \quad , \qquad \text{(III-16)}$$

and the load on the k'th element is

$$P_k = \frac{\theta}{2}(r_{ok}^2 - r_{ik}^2) \cdot \sigma_{yk} \quad .$$

Then the center of loading of the deformation unit may be determined by:

$$r_{CL} = \frac{\sum_{k=1}^{n} r_{ck} \cdot P_k}{\sum_{k=1}^{n} P_k} \quad . \qquad \text{(III-17)}$$

σ_{yk} is obtained by either of Equations (III-5) or (III-9) as the case may be either axisymmetric lateral flow between parallel platens or between inclined platens.

APPENDIX IV

<u>DEVELOPING A SURFACE FROM A SERIES OF PLANES</u>

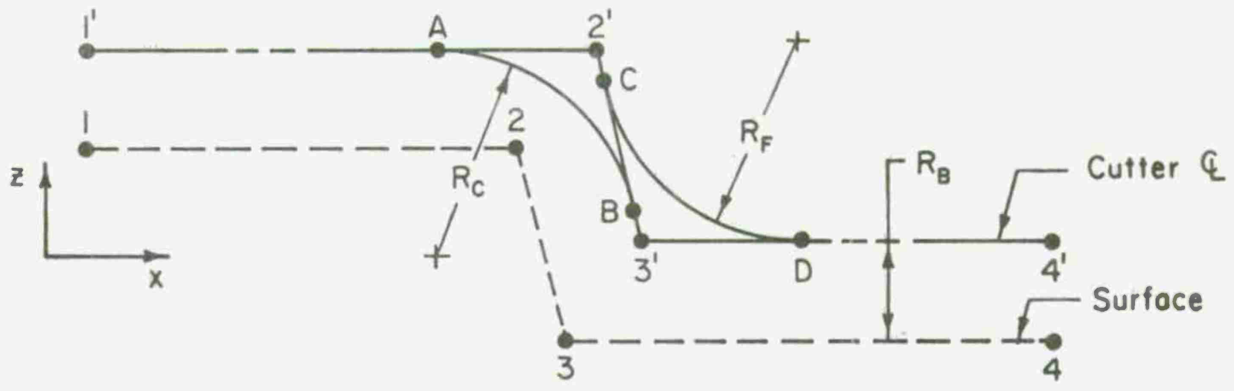
APPENDIX IV

DEVELOPING A SURFACE FROM A SERIES OF PLANES

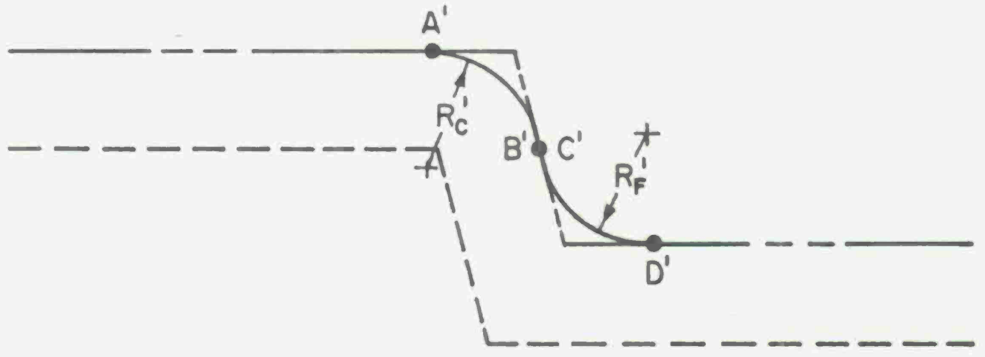
A solid surface, contoured in three dimensions, may be approximated by stacking a series of thin planes on top of each other. An example of this is an architectural model of a piece of land where pieces of cardboard are cut to the shape of the land contour at uniform steps in elevation. After each piece is located in proper relationship to its neighbor, the result is a three-dimensional approximation to the true piece of land. The smaller the elevation steps which each individual piece represents, the closer the model represents the actual item. When such an architectural model is made, horizontal planes are used.

A similar approach can be used to develop a model of a forging. However, in this case, vertical rather than horizontal planes are generally available. These vertical planes represent cross sections instead of constant elevation contours. The degree to which the model approximates the actual part, however, again depends on the number of sections used. The greater the number of sections, the better the model. Taking a series of slabs of appropriate thickness for each section, machining each to the appropriate contour, and then stacking them together is one approach to making a forging model. The major problem, however, is that the transition from one section to the next is an abrupt, vertical change.

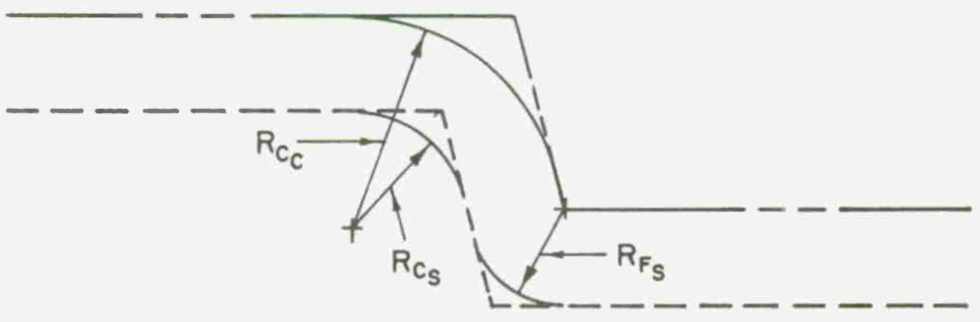
An improvement on this separate slab approach is to define each section numerically in one plane (i.e., X-Z) as a slab of some thickness and then generate numerical blends at the transition between each pair of slabs in a series of perpendicular planes (i.e., Y-Z). This procedure produces a surface composed of a large number of closely spaced slabs. This surface may then be manufactured using NC machining techniques. Several techniques for blending one slab into another were developed and evaluated. The one which is included in the track-shoe system, as part of subroutine PRTSRF, is as follows, referring to Figure IV-1.



A. Initial Trial



B. After Elimination of Cusps



C. After Compensation for Cutter Size

FIGURE IV-1. GENERATION OF SURFACE BLEND RADII

Given four points defining two adjacent points on a polygon, a trial radius for each point is developed by determining the length of each line segment adjacent to the point. The shorter of the two lines is selected as the trial radius for each point. The two radii are then evaluated for their tendency to form a cusp, rather than a smooth tangent. If they do cusp, they are both reduced in equal proportions until the cusp is eliminated.

Points 1, 2, 3, 4 define the surface of a polygon and 1', 2', 3', 4' define the offset cutter path polygon for a ball-end mill of radius R_B. Length 2'-3' is the shorter length for both points 1', 2', 3' and 2', 3', 4'. Therefore, the trial radius is set such that $R_F = R_C = 2' - 3'$. These radii form the cusp A, B, C, D, shown in Figure IV-1a. R_F and R_C are then reduced by equal amounts so that the tangent points B and C of the radii become coincident. This is shown in Figure IV-1b.

The resulting radii, R_C' and R_F' are then further modified by adding the cutter radius to a corner radius and subtracting the cutter size from a fillet radius. Thus, in Figure IV-1c

$$R_{F_c} = R_F' - R_B \tag{IV-1}$$

$$R_{C_c} = R_C' + R_B \quad . \tag{IV-2}$$

The reason for modifying the radii in this manner can best be seen by referring to Figure IV-2. This shows how a convex and concave surface is generated by a ball-end cutter. In this case $R_F = R_c = R_B$. Because the cutter radius, R_B, is equal to the corner radius, R_c, the locus of the points described by the surface of cutter becomes the corner point, A, itself. For the fillet, the locus becomes the radius of the center of the cutter plus the radius of the cutter. The result is that the surface radii become

$$R_{F_S} = 0 \tag{IV-3}$$

$$R_{C_S} = 2\ R_B \quad . \tag{IV-4}$$

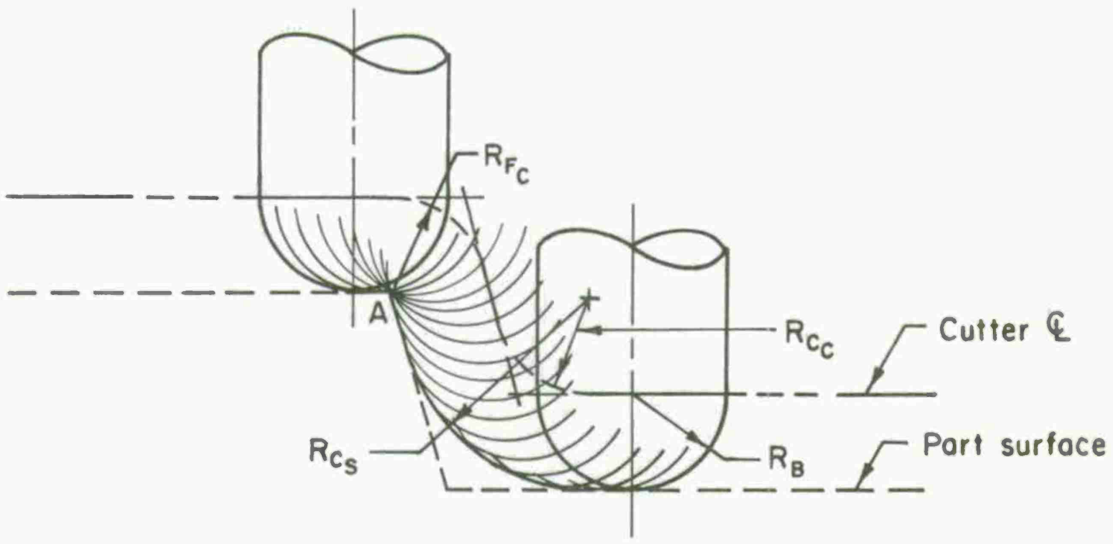


FIGURE IV-2. SURFACE GENERATED BY BALL-END MILL WHEN CUTTER SIZE EQUALS FILLET AND CORNER RADII

By applying Equations IV-1 and IV-2, the radii of the center of the cutter become

$$R_{F_c} = 0 \tag{IV-5}$$

$$R_{C_c} = 2\, R_B \quad , \tag{IV-6}$$

and the radii of the surface become

$$R_{F_s} = R_B \tag{IV-7}$$

$$R_{C_s} = R_B \quad . \tag{IV-8}$$

Equations IV-5 through IV-8 are for the specific case of $R_F = R_c = R_B$.

In the general case, the radii of the center of the cutter are

$$R_{F_c} = R_F' - R_B;\ R_{F_c} \geq \phi \tag{IV-9}$$

$$R_{C_c} = R_C' + R_B \quad , \tag{IV-10}$$

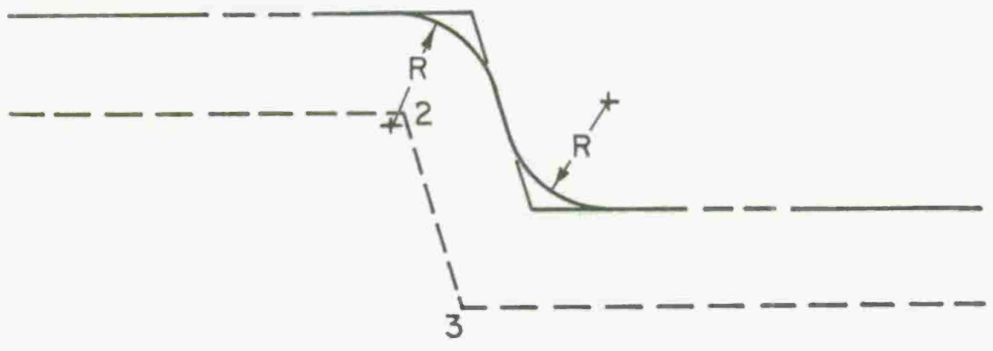
and the radii of the generated surface are

$$R_{F_s} = R_F + R_B = R_F' \tag{IV-11}$$

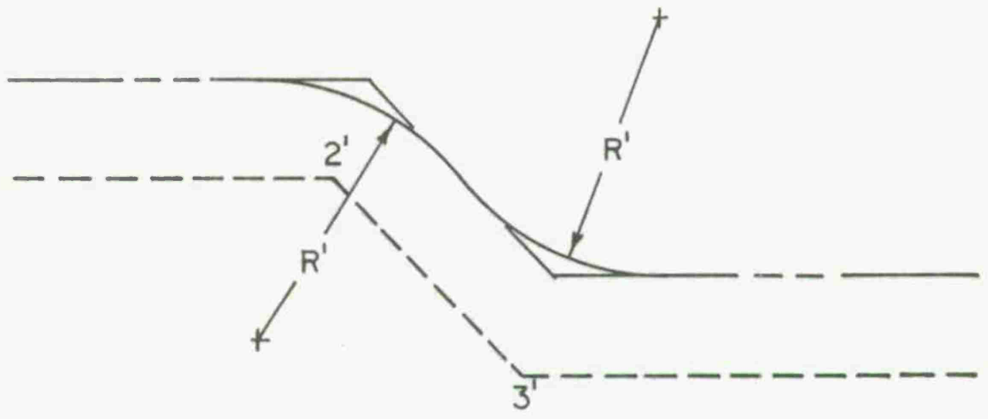
$$R_{C_s} = R_{C_c} - R_B = R_C' \quad . \tag{IV-12}$$

Referring to Figure IV-2, it should be noted that the surface generated when $R_F = R_B$ is the same whether the cutter center moves along the arc defined by R_F or along the straight-line path which defines the corner of the polygon.

As a result of using Equation IV-9, it is possible to have a negative value for R_{F_c}. If this happens, R_{F_c} is set to zero. The resulting values for R_{F_c} and R_{C_c} are tested for cusps, and reduced in size as necessary.



A. Small Draft



B. Large Draft

FIGURE IV-3. EFFECT OF POLYGON POSITION (DRAFT ANGLE) ON BLEND RADII

It should be noted that the surface generated by the above is a result of the cutter size specified, the algorithm used to eliminate cusps, and the polygon itself. Larger blend radii in the machined surface could also be achieved by increasing the draft, or by modifying the Y-dimension given by the user at the time the section planes are put into sequence. Both of these techniques have the same effect. That is, referring to Figure IV-3, they would tend to shift Point 2 to the left and Point 3 to the right. Because of this shift, the radii which result, R', after checking for cusps the first time and before applying Equations IV-1 and IV-2, are larger than they were with the small draft polygon.

APPENDIX V

ENERGY REQUIRED IN CLOSED-DIE FORGING

APPENDIX V

ENERGY REQUIRED IN CLOSED-DIE FORGING

The energy required to produce a forging is the area under the load-displacement curve. A typical load-displacement curve is shown in Figure V-1. With load expressed in pounds and displacement in inches, the energy is units of inch/pounds. Experimentally, the energy may be found using an X-Y recorder. Generally, the displacement value, obtained from a linear potentiometer, LVDT, or similar device is recorded on the X axis. The load value is obtained from strain bars on the press columns and is recorded on the Y axis. The energy is then found using a polar planimeter or by numerical integration.

Energy is not a significant consideration for hydraulic presses as they can generate maximum load at any stroke position. As long as the load generated by the press is greater than that required by the forging, the forging will be completed. If the press capacity is less than required, the press will stall without completing the required deformation.

Mechanical (eccentric or crank) presses are stroke-restricted devices since the load developed is a function of the ram position. Screw presses, hammers, and High-Energy Rate Forming (HERF) machines are energy-restricted devices. When forging in a mechanical press, if the energy required equals or exceeds that available, the press will slow down by an unacceptable amount, thereby limiting the production rate. In the extreme case, the press may stall. For energy-restricted devices, if the energy required exceeds that available, more than one stroke will be required to produce the part.

To calculate the energy required for a forging analytically, it would be necessary to calculate the load on each section for each increment of stroke. Due to the complexity of the parts considered in this program, with their many cavities, calculating the load at other than full closure appears to be an impossible task. If possible at all, making such a calculation would require an extreme amount of computer time.

Because of the problems in determining the energy required for the real forging, a simple model of the forging die cavity is generated. This is shown in Figure V-2. The height and width of the linear portion are given by:

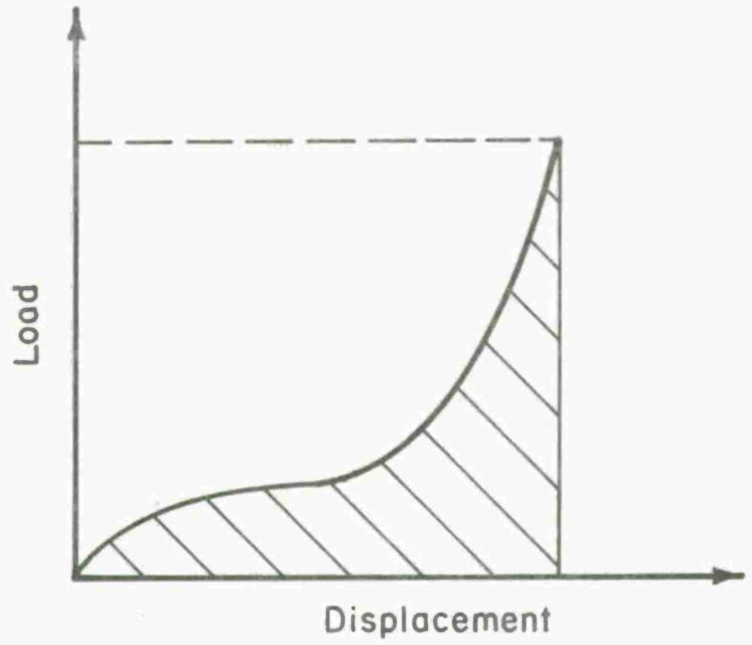


FIGURE V-1. TYPICAL LOAD-DISPLACEMENT CURVE FOR CLOSED-DIE FORGING WITH FLASH

$$\bar{H} = \text{Total Volume/Total Plan Area} \tag{V-1}$$

$$\bar{W} = \Sigma\ (W_I * D_I)/\Sigma\ D_I \quad , \tag{V-2}$$

where W_I = the width of section I, excluding flash

D_I = depth of section I (plane strain)

= $\theta . R_{CG}$ for section I (axisymmetric strain).

As shown in Figure V-2, the model has a linear-center section which is taken to be in plane strain. At each end, there is a semicircular section which is considered to be in axisymmetric strain. The perimeter of the section is entered by the operator subject to the condition that:

$$P \geq 2\ \Sigma\ D_I + \pi\ .\ \bar{W} \quad . \tag{V-3}$$

From this, the depth of the model is found as:

$$\bar{D} = (P - \pi\ \bar{W})/2 \quad . \tag{V-4}$$

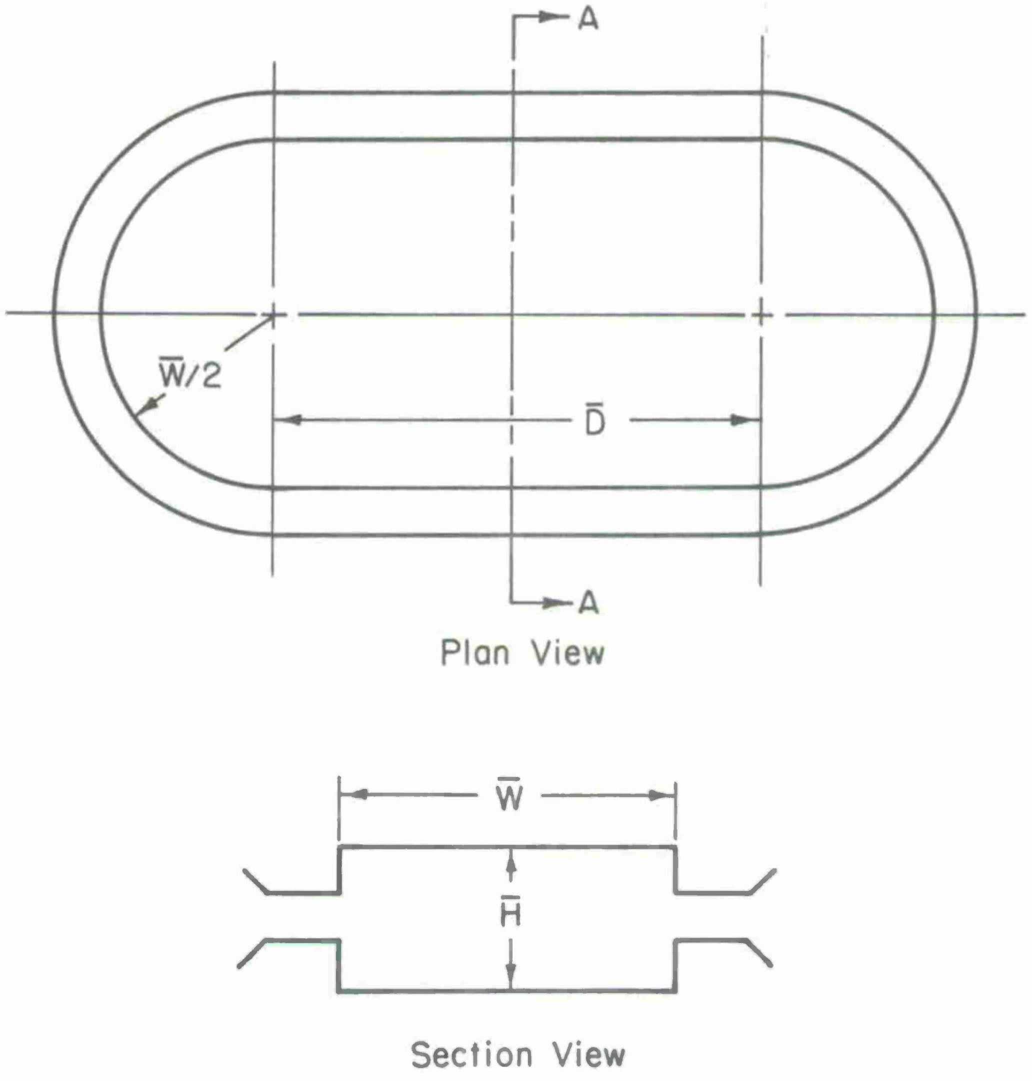


FIGURE V-2. ENERGY MODEL DIE CAVITY

The inequality in Equation V-3 is used to account for irregularities in the perimeter of the actual part. In the track shoe considered in this program, there were several places where one section was offset from another, thus adding considerably to the perimeter. Because load is considerably influenced by the amount of flash and the amount of flash is directly related to the perimeter, it is necessary to model the amount of flash as accurately as possible.

The three dimensions, $\bar{W}$, $\bar{H}$, and $\bar{D}$ which are used to describe the cavity of the model, are thus all derived as weighted averages from the actual part. The flash height and width in the model are taken as those used in determining the load for each section.

The billet area to be used in the model is chosen as the maximum cross section of any of the sections analyzed. To this is added a percentage for flash and other losses. The billet is assumed to have a square cross section and thus has a side dimension of

$$B = (\text{Section Area}_{\text{max}})^{1/2} . \qquad \text{(V-5)}$$

Referring to Figure V-3, the ends of the model billet are taken as square with a total volume of B^3 for the two ends. This is to ensure sufficient material is available to fill the axisymmetric cavity. In the load calculations on the ends, however, the ends are assumed to be cylindrical and that the radial deformation is uniform at all positions.

Starting with the original billet size, the energy model calculates the load on the plane-strain cavity at seven stroke positions. The seven positions, shown in Figure V-4, are as follows:

(1) Dies in contact with billet and loaded so that deformation is just about to start

(2) Deformation such that forging width, FW, = $B/2 + .25\ (\bar{W}/2 - B/2)$

(3) $FW = B/2 + .5\ (\bar{W}/2 - B/2)$

(4) $FW = B/2 + .75\ (\bar{W}/2 - B/2)$

(5) $FW = \bar{W}/2$

(6) Cavity filled and flash extruded to edge of flash land

(7) Forging completed; forging height = $\bar{H}$.

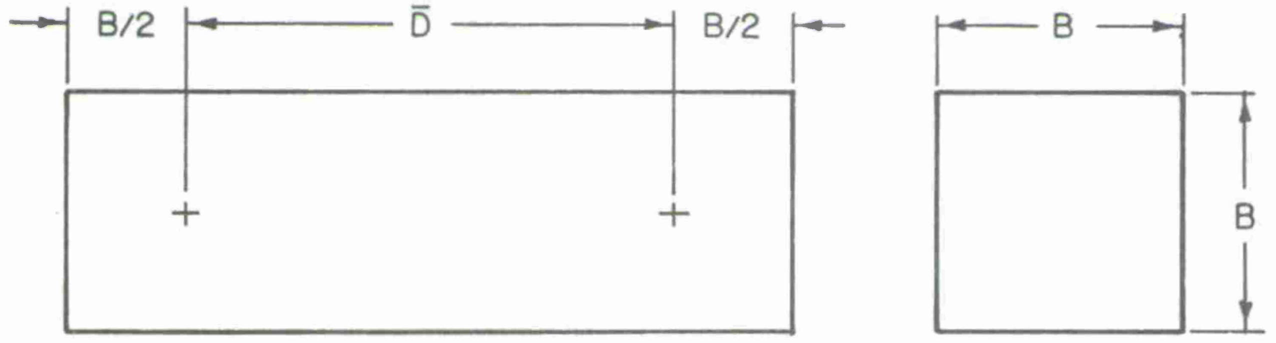


FIGURE V-3. ENERGY MODEL BILLET

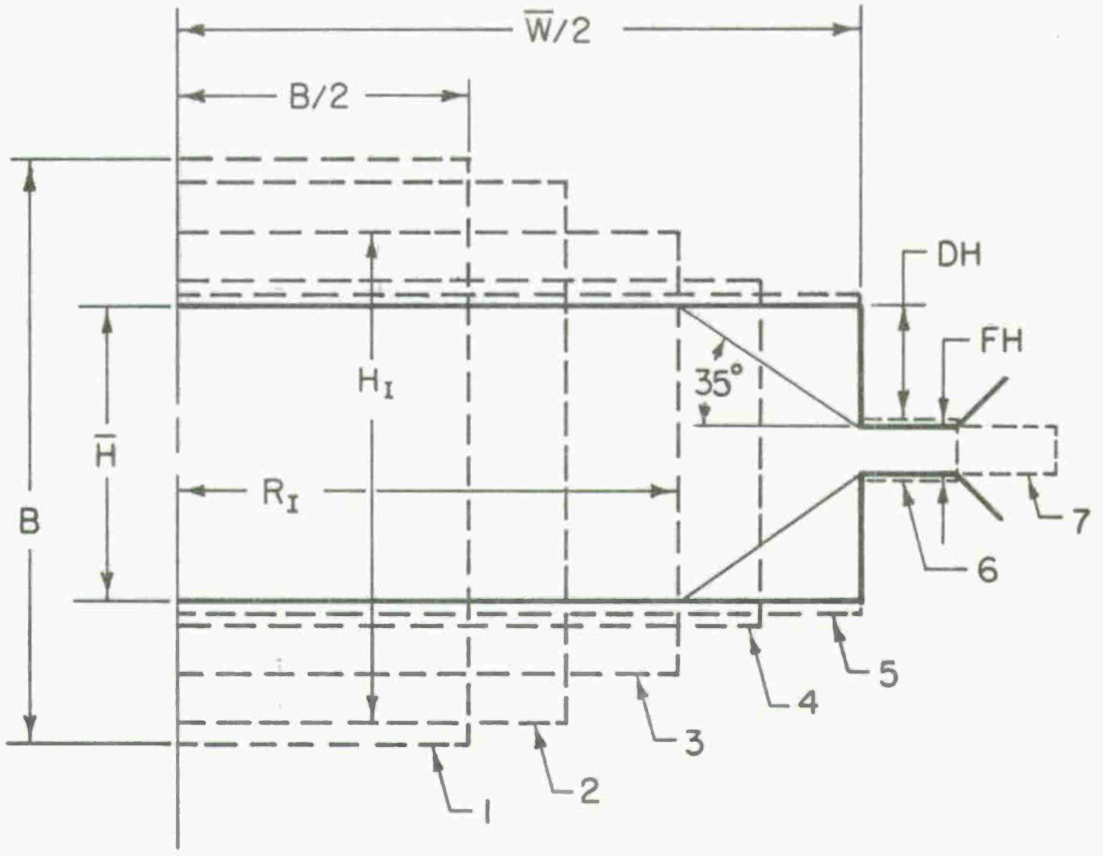


FIGURE V-4. BILLET POSITIONS FOR SEVEN-STEP PLANE-STRAIN ENERGY MODEL

By knowing the original billet size and fixing the forging width at each step as above, the height of the forging can be found by maintaining volume constancy. The displacement at each step is the difference between the original billet height and the forging height. The flow model assumes the free surface of the billet and remains as a vertical plane throughout the forging process (no bulging). Strain rate and temperature effects are also neglected.

For the plane-strain cavity, the model stays in friction flow up to and including Step 4. That is, through Step 4, the forging is assumed to be totally within the model cavity with no shear. Using Equations III-1 and III-2, the beginning stress on the forging is found as

$$\sigma_{yb} = 2\ (f)\ (FS)\ (WIDTH/HEIGHT) + \sigma_{ye} \quad , \qquad (V-6)$$

where f = friction factor (entered by user)

FS = material flow stress in plane strain

$= (2/\sqrt{3})\ \sigma$.

Using the value for σ_{yb} found in Equation V-6, the load on the forging is found as:

$$P = .5(\sigma_{yb} + \sigma_{ye})\ (WIDTH) \ . \qquad (V-7)$$

When Step 5 is reached, the forging is analyzed as two elements. One element is in shear and one element is in friction. The shear angle is assumed to be 35 degrees. The dimensions of the element in shear are:

Beginning Height = $FH_I - 2\ DH$

where $DH = .5(\bar{H} - FT)$

Ending Height = FH_I

Width = DH/tan 35

Using Equations III-3 and III-4, and taking $\alpha = \beta = -35$ (tan 35 = .7002) and $f = 1/\sqrt{3} = .577$,

$$K_1 = \tan\alpha + \tan\beta \qquad (V-8)$$

$$= -1.4004 \qquad (V-9)$$

$$K_2 = -\frac{2}{\sqrt{3}}\bar{\sigma} K_1 + f\,\bar{\sigma}\,(2 + \tan^2\alpha + \tan^2\beta) \quad \text{(V-10)}$$

$$= \frac{\bar{\sigma}}{\sqrt{3}}(-2K_1 + (2 + 2\tan^2\alpha)) \quad \text{(V-11)}$$

$$= .577\bar{\sigma}\,(-2(-1.400) + (2 + 2(.490))) \quad \text{(V-12)}$$

$$= .577\bar{\sigma}\,(2.800 + 2.980) \quad \text{(V-13)}$$

$$K_2 = 3.337\bar{\sigma} \quad \text{(V-14)}$$

$$K_2/K_1 = \frac{3.337\bar{\sigma}}{-1.400} \quad \text{(V-15)}$$

$$= -2.383\bar{\sigma}\ . \quad \text{(V-16)}$$

Equations III-3 and III-4 are used to find the beginning stress and load on the shear element. The beginning stress on the shear element is then used as the ending stress on the plane-strain element which fills the remainder of the cavity.

For Steps 6 and 7, a third element is added. This is the material in the flash and is in plane strain. The total load at each step is the sum of the loads on each element in use at that step. Figure V-5 depicts the fully developed, three-element energy flow model. The loads calculated are for a unit depth and for half of the total model cavity. Therefore, it is necessary to multiply the load at each step by $2\bar{D}$.

For the ends of the model which are in axisymmetric strain, the analysis is reversed. For the seven steps of the axisymmetric case, the displacements calculated for the plane-strain cavity are used as the governing factor. Knowing the displacement at each step and maintaining volume constance, the radius of the forging can be found. Once the radius exceeds $\bar{W}/2$, the material extruded into the flash must be determined.

From volume constancy,

$$B^3 = 2\pi\,(\bar{W}/2)^2\,DH + \pi\,R^2\,(H_I - 2\,DH)\ . \quad \text{(V-17)}$$

Solving for R gives:

$$R_I^{\,2} = [B^3 - 2\pi\,(\bar{W}/2)^2\,DH]/\pi\,(H_I - 2DH)$$

where $DH = .5(\bar{H} - FT)$

FT = Flash thickness

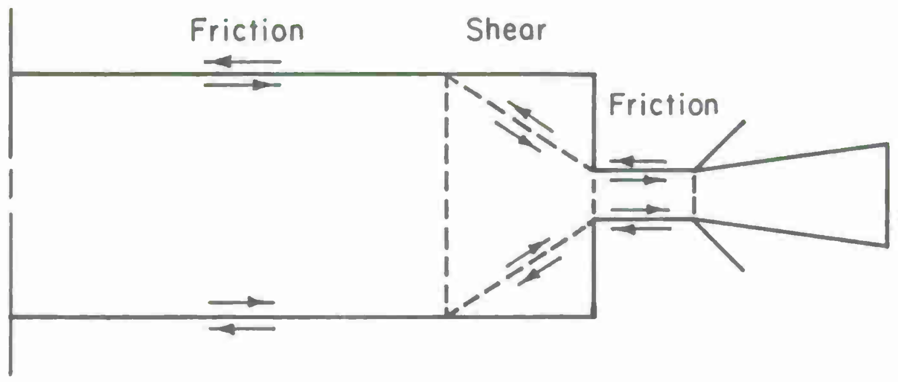


FIGURE V-5. THREE-ELEMENT ENERGY FLOW MODEL

H_I = Forging height

$= B - D_I$

D_I = Displacement at Step I.

Since the numerator of Equation (V-10) is a constant, R_I can be expressed as:

$$R_I = [K/\pi\ (H_I - 2 * DH)]^{1/2} \qquad \text{(V-19)}$$

where

$$K = B^3 - 2\pi\ (\bar{W}/2)^2\ DH \qquad \text{(V-20)}$$

The stress and load analysis for the axisymmetric section proceeds in a similar fashion to the analysis for the plane-strain cavity. Friction flow is assumed for the flash area, shear flow for the cavity element next to the flash, and a final friction flow element to complete the cavity.

APPENDIX VI

<u>COMPUTER HARDWARE CONFIGURATION, OPERATING SYSTEM SOFTWARE, AND PROGRAM ORGANIZATION</u>

APPENDIX VI

COMPUTER HARDWARE CONFIGURATION, OPERATING SYSTEM SOFTWARE, AND PROGRAM ORGANIZATION

Computer Hardware System

The complete forging die-design system has been developed and implemented on a Digital Equipment Corp (DEC) PDP-11/40 computer system. This computer is composed of the following elements:

(1) 32K (16-bit word) core memory. The upper 4K of memory is used for peripheral device registers leaving 28K available for user programs.

(2) High-speed paper-tape reader and punch.

(3) Two RK-11 disk-pack drives (1.2 million words each).

(4) LA-30 DEC-writer keyboard terminal.

(5) VT-11 refresh graphics display and control, including light pen.

(6) LPS data acquisition system. The analog output of this unit is used to drive the X-Y recorder for hard-copy plots.

(7) Hewlett Packard Model 7004B flat-bed X-Y recorder.

A special interface was built to adapt the LPS output control signals to the X-Y recorder.

Operating System Software

The programs were developed and operate using the following DEC software:

(1) RT-11 single-job operating system, version VØ2B-Ø5E

(2) FORTRAN-IV compiler, VØ1B - Ø80

(3) Link editor, VØ4 - Ø2

(4) MACRO assembler

(5) VTHDLR graphics driver. This is loaded with the graphics handlers at link time to generate the graphic displays and handle the interaction with the light pen.

The entire system of programs has been written in FORTRAN-IV with the exception of the graphics display and plotter handlers, which are in MACRO assembler. The program system is linked as one root segment with two overlay regions. The upper core address used by the programs (excluding the RT-11 operating system) is approximately 20761 (base 10 words). The two overlay regions combined use 3651 words; the combined length of the 25 segments which run in these two regions is approximately 28,868 words. The organization of the program modules is given in Table VI-1, with the run time memory organization shown in Figure VI-1. The program organization is shown graphically in the drawing attached to this report.

Compiling and linking is performed following the directions in DEC manual No. DEC-11-LRFPA-A-D, "RT-11 FORTRAN Compiler and Object Time System User's Manual". When compiling, module TRACK should be the last module compiled, and it should be the only module which uses the "U" compiler switch.

Linking the object modules to create the save-image file with the overlay structure is normally done using the BATCH processor of RT-11. The link commands for BATCH are contained in file LNKTRK.BAT, and are listed in Figure VI-2. The file FORTRK.BAT is available for BATCH compiling all program segments.

Before running TRACKS, the monitor directive "GT ON" should be specified to indicate that the GT-40 display is to be used.

The overlay structure created by the linker is such that all Region 2 overlays are positioned above the top of the largest segment in the Region 1 overlay area. Thus, the total overlay area required is the sum of the largest segments in each region. From Table VI-1, it can be seen that total core required for overlays is governed by segments PREFRM in Region 1, and CADSB1 in Region 2. Furthermore, it has been found by experience that the upper limit of 20,7611 (base 10 words; 121062 base 8 bytes) is very close to the maximum allowable program core-load size. Care should be taken in modifying these segments to ensure that the segments do not become so large as to prevent loading and execution.

TABLE VI-1. ORGANIZATION OF PROGRAM MODULES

Overlay Region No.	Segment No.	Segment Name	Segment Size(1)	Module Name	Module Type(2)	Code(3)
Ø	1	TRACK	8480	TRACK	P	F
				Blank Common	C	F
				SYSPR	C	F
				SECTN	C	F
				DISPLA	C	F
				STRSPT	C	F
				DEFEL	C	F
				FINISH	C	F
				MAXMIN	S	F
				XYINTR	F	F
				SCALZZ	S	F
				ACOS	F	F
				TAN	F	F
				CENTER	S	F
				MARKIT	S	F
Ø	1	GRPHCS	842	GT40	G	M
				GT40TC	G	M
				VTHDLR(*)	G	M
1	1	PRPROS	1252	PRPROS	S	F
				RDIOF4	S	F
				ROTATE	S	F
				PARAMS	S	F
1	2	PICTUR	1174	PICTUR	S	F
				DIESRF	S	F
				INITDS	S	F

TABLE VI-1. (Continued)

Overlay Region No.	Segment No.	Segment Name	Segment Size(1)	Module Name	Module Type(2)	Code(3)
1	3	STRESS	1856	STRESS	S	F
				DBLFST	S	F
1	4	ERRPRT	942	ERRPRT	S	F
				SETUPS	S	F
1	5	FLWSRF	1353	FLWSRF	S	F
				SHRBND	C	F
1	6	PFPREP	1829	PFPREP	S	F
				GETRID	S	F
1	7	PREFRM	2330	PREFRM	S	F
				SORT	S	F
1	8	SEQSRF	1130	SEQSRF	S	F
				WRITIT	S	F
1	9	PRTSRF	2302	PRTSRF	S	F
				REDUCE	S	F
				PUNCHO	C	F
1	10	ENERGY	1391	NERGY	S	F
				PRLFLW	F	F
				PLNSLD	S	F
				AXYSLD	S	F
2	1	OVRLY4	1162	ADDPTS	S	F
				SHRPTS	S	F
				FORCE	S	F
2	2	OVRLY5	813	MERGE	S	F
				FILTER	S	F
2	3	OVRLY6	1043	XSECA	S	F
				VOLUME	S	F
2	4	OVRLY7	576	INTRPL	S	F
				FITARC	S	F
				RIBRAD	S	F
				RIBFIL	S	F

TABLE VI-1. (Continued)

Overlay Region No.	Segment No.	Segment Name	Segment Size(1)	Module Name	Module Type(2)	Code(3)
2	5	OVRLY8	1053	DECUSP	S	F
				INT2LN	S	F
				LINEQ	S	F
				RADEXP	S	F
				MOVEND	S	F
				YOFFST	S	F
2	6	OVRLY9	1253	GETPTS	S	F
				RESLTS	S	F
				SAVPLY	S	F
2	7	DEFREL	510	DEFREL	S	F
2	8	PLSTRS	839	PLSTRS	S	F
2	9	AXSYUF	753	AXSYUF	S	F
2	10	CADSB1	1320	GETHIT	S	F
				RIBID	S	F
				RIBPRM	S	F
				ADDRIB	S	F
2	11	CADSB2	742	ADDPT	S	F
				NEWRIB	S	F
2	12	PLOTER	391	PLOTER	S	F
2	12	PLOTOB	558	PLOTOB	G	M
2	13	CAMSB1	872	CLPTS	S	F
				DELETE	S	F
				AITKEN	S	F
				LINEAR	S	F
2	14	SAVENC	576	SAVENC	S	F

TABLE VI-1. (Continued)

Overlay Region No.	Segment No.	Segment Name	Segment Size(1)	Module Name	Module Type(2)	Code(3)
2	15	PNCHNC	898	PNCHNC	S	F
				NCOUT	S	F
				PARTNO	S	F
				STOPNC	S	F
				PXYZ	S	F
				LEADER	S	F
				STORE	S	M
				OUTPUT	C	F

(1) Size in base 10 words

(2) Module Type - C = Common, F - Function, G = Group of FORTRAN callable subroutines, P = Program, S = Subroutine

(3) Code - F = FORTRAN, M = Macro Assembler

(*) Object code supplied by DEC with VT-11 display system hardware.

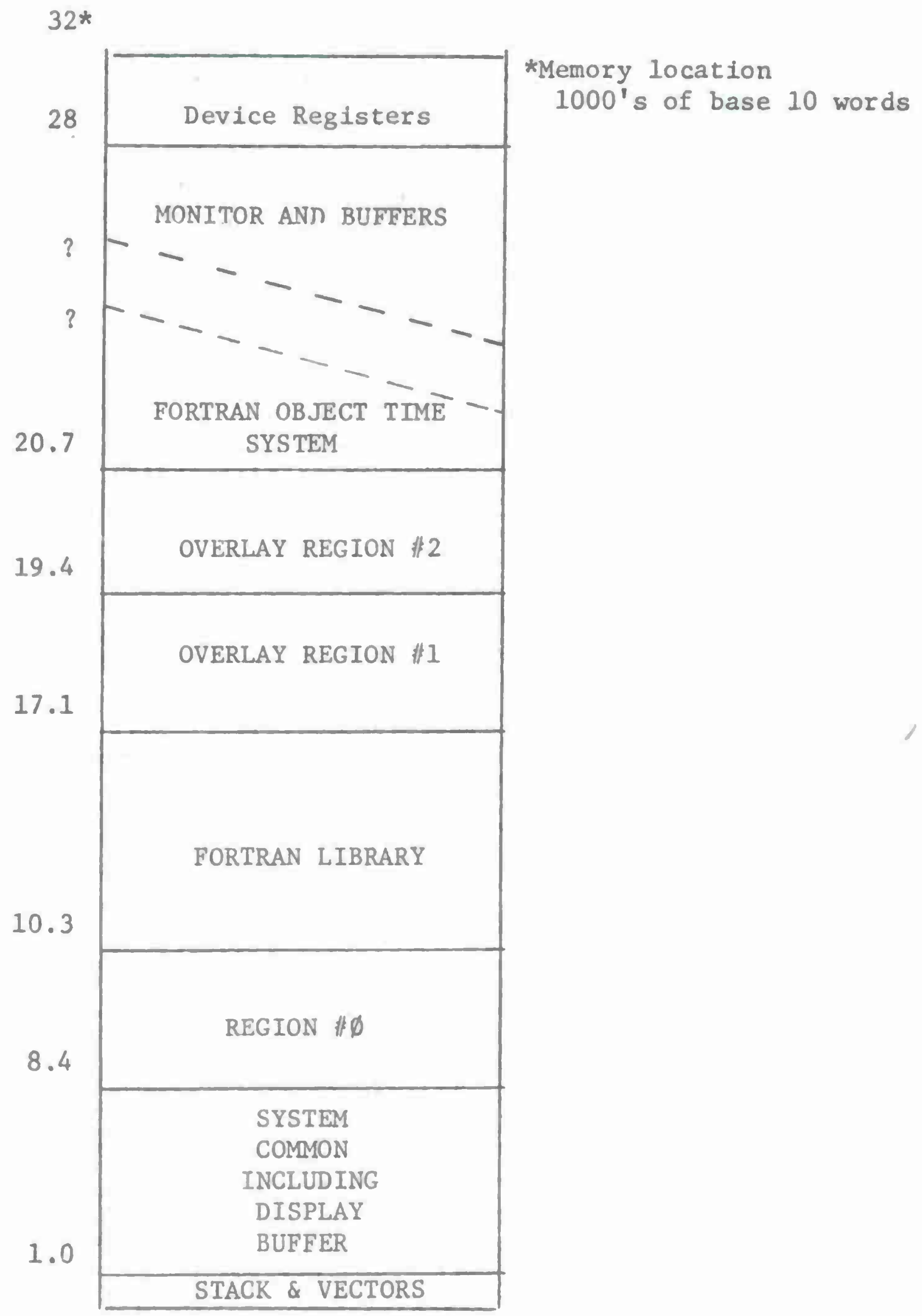


FIGURE VI-1. RUN TIME MEMORY ORGANIZATION

```
.R LINK
*TRACKS,TRACKS.MAP=TRACK,GRPHCS/F/B:1200/C
*PRPROS/O:1/C
*PICTUR/O:1/C
*STRESS/O:1/C
*ERRPRT/O:1/C
*FLWSRF/O:1/C
*ENERGY/O:1/C
*PFPREP/O:1/C
*PREFRM/O:1/C
*SEQSRF/O:1/C
*PRTSRF/O:1/C
*SAVENC/O:2/C
*OVRLY4/O:2/C
*OVRLY5/O:2/C
*OVRLY6/O:2/C
*OVRLY7/O:2/C
*OVRLY8/O:2/C
*OVRLY9/O:2/C
*DEFREL/O:2/C
*PLSTRS/O:2/C
*AXSYUF/O:2/C
*CADSB1/O:2/C
*CADSB2/O:2/C
*CAMSB1/O:2/C
*PNCHNC,STORE/O:2/C
*PLOTER,PLOTOB/O:2
```

FIGURE VI-2. BATCH FILE COMMAND STRING TO LINK EDITOR (LNKTRK.BAT)

```
.RUN FORTRA.SAV
*PRPROS=PRPROS
*PICTUR=PICTUR
*STRESS=STRESS
*ERRPRT=ERRPRT
*FLWSRF=FLWSRF
*ENERGY=ENERGY
*PFPREP=PFPREP
*PREFRM=PREFRM
*OVRLY4=OVRLY4
*OVRLY5=OVRLY5
*OVRLY6=OVRLY6
*OVRLY7=OVRLY7
*OVRLY8=OVRLY8
*OVRLY9=OVRLY9
*DEFREL=DEFREL
*PLSTRS=PLSTRS
*AXSYUF=AXSYUF
*CADSB1=CADSB1
*CADSB2=CADSB2
*SAVENC=SAVENC
*PLOTER=PLOTER
*CAMSB1=CAMSB1
*PNCHNC=PNCHNC
*SEQSRF=SEQSRF
*PRTSRF=PRTSRF
*TRACK=TRACK/U
```

FIGURE VI-3. BATCH FILE COMMAND STRING TO FORTRAN COMPILER (FORTRK.BAT)

APPENDIX VII

DESCRIPTION OF SUBROUTINES

APPENDIX VII

DESCRIPTION OF SUBROUTINES

Function ACOS(X)

This function returns the arc cosine of X, computed by the expression

$$acos(X) = atan\left((1 - X^2)^{.5}/X\right) \quad . \tag{VII-1}$$

Because the square root routine may generate an error when computing the square root of Ø, tests are made for X = 1 and X = -1. When these conditions are found, the arc cosine is assigned the value Ø, or π, respectively.

Subroutine ADDPT(J1)

ADDPT is used to add a point to the preform polygon wherever indicated by the designer. J1 is the X-coordinate (in CRT raster units) where the point is to be located. The surface hit (upper or lower) is passed through common as K1. The routine first assumes the hit was on the upper surface and then tests K1 to verify this assumption. Search indices are set as appropriate for the surface indicated. The coordinate of the hit is transformed back into user dimensions, and the proper section of the preform polygon array is then searched to find the two existing points which bound the new point. The Z-coordinate of the new point is found from the two boundary points by using function XYINTR. The preform polygon array is then opened so the new point can be inserted at the proper position.

The designer is then asked if the new point is to be a permanent addition. If he types "Y" (yes), he is then asked to indicate the sense (fillet or corner) of the new point. This is used to append the correct sign to the radius value of the new point. The point is also offset, either to the inside or the outside, based on the sense indicated. Permanent points are assigned a radius of .25 inches and are located .1 inch away from the line between the boundary points.

If the point is not to be permanent, it is located exactly on the line between the two boundary points, and is given a radius of 10^{30}. This large radius is used by GETRID to eliminate the point during subsequent processing.

Data is passed to this subroutine by the argument J1, described above, and through Labeled Common blocks DEFEL, DISPLAY, SECTN, and SYSPR.

Subroutine ADDPTS (XB, YB, NB, XSRF, XI, YI, FLAG, TOP)

ADDPTS generates two points on the shear surface defined by two boundary points. It is called separately for the upper and lower surfaces. The following discussion will be in terms of adding points with shear boundaries on the upper surface. Adding points with shear boundaries on the lower surface would be exactly the same, except the references to the surfaces would be reversed.

To find the new points for the upper shear boundaries, it is first necessary to find the points on the lower surface which lie directly below the shear boundaries. This is done by scanning the lower surface for a set of X values which bracket the X value of each shear boundary. When such a set is found, the Y value is determined by linearly interpolating between the Y values associated with the points bracketing the shear boundary. If the section is single flash axisymmetric with a shear boundary on the axis of symmetry, the two points which bracket the first shear boundary describe a vertical line. In this case, the Y value for the lower surface point is taken as the average of the Y values.

When all the points opposite shear boundaries have been found, a test is made to determine if the part is in axisymmetric, single flash strain with a shear boundary on the axis of symmetry. If it is not, the following operations are skipped. If these conditions are met, a mirror image of the first cavity is generated. This is done since the part is axisymmetric; the cavity and the part actually lie on both sides of the axis, even though only the right half is being analyzed. The X and Y coordinates for the left shear boundary and its opposite point on the lower surface are then set equal to the mirror image of the right shear boundary and its associated opposite point (see Figure VII-1).

The points on the shear surface are then found by calling subroutine SHRPTS, and passing the coordinates of the shear boundaries and the points opposite the shear boundaries on the lower surface. It should be noted that the variables XI and YI, used to pass the coordinates of the opposite points to SHRPTS, contain the coordinates of the shear surface points on the return from SHRPTS.

If the part is not axisymmetric, single flash with a shear boundary on the axis of symmetry, upon returning from SHRPTS, the subroutine is exited with control returned to FLWSRF. If these conditions are met, the first shear boundary and the first shear surface point are set equal. They are both given X coordinates equal to the axis of symmetry, and Y coordinates equal to the Y coordinate of the second shear surface point. Control is then returned to FLWSRF.

Calling sequence arguments:

XB, YB:	Arrays of coordinate points defining the location of shear boundaries (input).
NB:	The number of shear boundaries plus 1 (input)
XSRF, YSRF:	Arrays of coordinate points defining the surface opposite the surface on which the shear boundaries lie (input).
NSRF:	The number of points defining the surface (input).
XI, YI:	Arrays for coordinates of the points calculated to lie on the shear surface (output).
FLAG:	A logical variable used to indicate that the section is in axisymmetric, single flash strain with a shear boundary on the axis of symmetry (input).
TOP:	A logical variable used to indicate that the shear boundaries lie on the upper part surface. This is not used by the current version of SHRPTS (input).

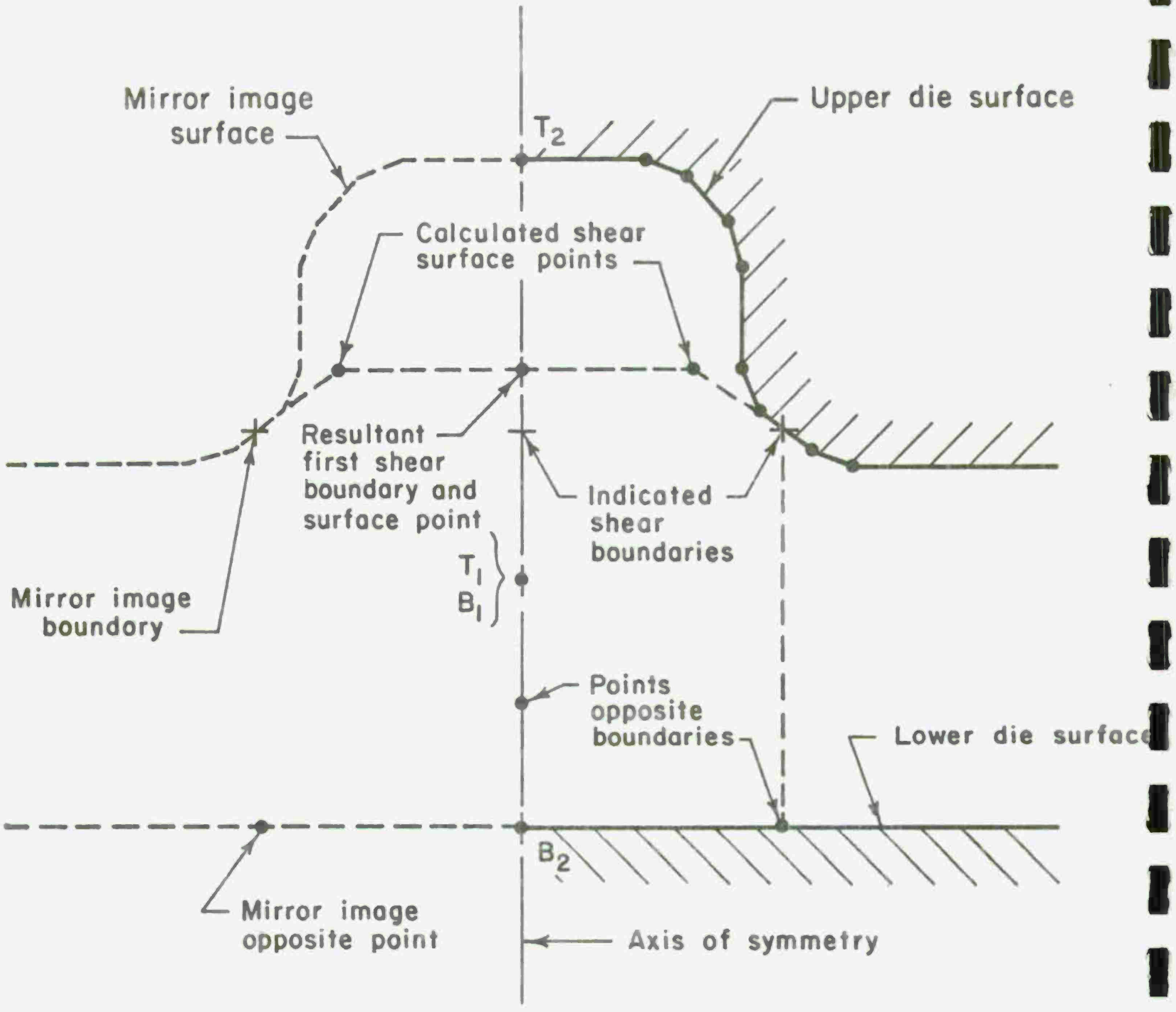

Mirror image surface
Upper die surface
T_2
Calculated shear surface points
Resultant first shear boundary and surface point
Indicated shear boundaries
T_1
B_1
Mirror image boundary
Points opposite boundaries
Lower die surface
B_2
Mirror image opposite point
Axis of symmetry

FIGURE VII-1. GENERATION OF SHEAR SURFACE POINTS WHEN A CAVITY IS BOUNDED BY THE AXIS OF SYMMETRY

Subroutine ADDRIB (ITYPE,PFRMPL)

This subroutine generates the four coordinates needed to define a new rib. Its primary purpose is to allow a new rib to be added opposite an existing rib to prevent suck-in defects.

When called, it first uses RIBID to locate all existing ribs on the part. If there are none, it exits. If one or more ribs exist, the first corner of each rib is displayed as a light-sensitive point. The designer is then asked to indicate the existing rib of interest. The new rib will be added opposite this rib, if possible. After the rib of interest is indicated with the light pen, the opposite surface is tested to determine that the surface is a single web. This is done by checking all points on the opposite surface to see that the X coordinates of the two fillets of the rib are bounded in X by the same pair of points on the opposite surface.

Referring to Figure VII-2, the rib fillet points 3 and 6 are both bounded by the same pair of opposite points, 13 and 14. The general form of the test is as follows:

Let XF1 and XF2 be the X coordinates of the rib fillets. Let XL and XM be the X coordinates of two adjacent points on the opposite surface. Then let

$$A = (XF1 - XL) * (XF1 - XM) \qquad \text{(VII-2)}$$

$$\text{and} \quad B = (XF2 - XL) * (XF2 - XM). \qquad \text{(VII-3)}$$

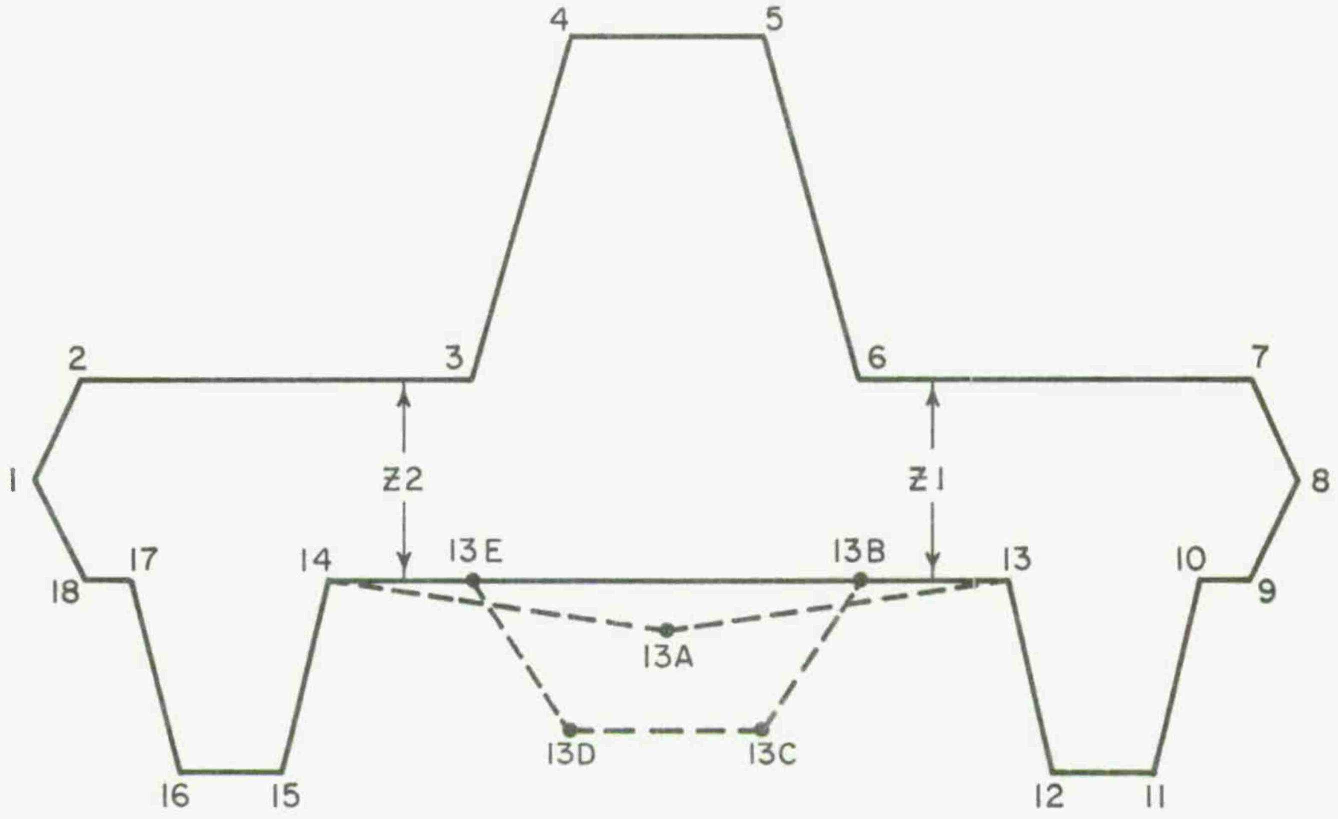
If both A and B are negative, XL and XM bound XF1 and XF2. Referring to Figure VII-2, this criterion would not be satisfied if Point 13A were part of the lower surface. If a single surface opposite the indicated rib cannot be found, a message to this effect is typed, and the subroutines exited.

If a new rib can be added, subroutine RIBPRM is used to obtain the parameters of the existing rib.

The thicknesses, Z1 and Z2, between the existing rib fillets and the opposite surface are then determined. The height of the new rib is then equated to

$$\text{New Height} = \frac{\text{K (Existing Rib Height)(Existing Rib Ratio)}}{\text{(Average Web Thickness)}}, \qquad \text{(VII-4)}$$

where K is a constant. That is, the height is set directly proportional to



Subroutine ADDRIB

FIGURE VII-2. ADDITION OF A NEW RIB OPPOSITE AN EXISTING RIB

the existing rib height and the height-to-width ratio of the existing rib, and inversely proportional to the average web thickness.

The next operation is to open up the coordinate arrays so that the four points defining the new rib can be added. In Figure VII-2, this would have the effect of changing points 14-18 to points 18-22.

The coordinates of the four new points are found as follows. The X coordinates of the new fillets are equated to the X coordinates of the existing rib at the point where the bottom rib width is determined (XS and XT in Figure VII-29). The Y coordinates of the new fillets are the intersection of the X coordinates with the web opposite the existing rib. The horizontal distance between each new fillet and its adjacent new corner is half the difference between the top and bottom widths of the existing rib. When the existing rib is perfectly symmetrical, as in Figure VII-2, the X coordinates of the four new points will lie directly above, or below the four coordinates of the existing rib. The Y coordinates of the two new corners are the calculated new rib height below or above the web previously found opposite the existing rib. The radii for all four points is equated to half the top rib width, with the signs set as appropriate. In Figure VII-2, the four new points are shown as 13B, 13C, 13D, and 13E.

The final operation is to increment by four the counter of the number of points describing the section. If the new rib was added to the upper surface, the index to the right-hand parting line point is also incremented by four.

The parameters passed in the calling sequence are:

(1) ITYPE - A value indicating the type of section being operated on (plane or axisymmetric)(input).

(2) PFRHPL - The index to the right-hand parting line coordinate (input).

The following Named Common blocks are also used to transfer data: SYSPR, DISPLA, and DEFEL.

Subroutine AITKN (X,Y,N,K,XB,YB,TEMP,IEX)

AITKN is a subroutine for polynomial interpolation of a table of values of Y versus X. The user specifies the degree of the interpolating polynomial. If the argument falls outside the range of tabular values, extrapolation is performed.

Repeated bisection of the index (subscript) of the list of values of X is carried out until the argument is isolated between two consecutive values. Then Aitken's well-known recursion of linear interpolation is used. When possible, the routine interpolates from an equal number of tabular points on either side of the argument. When the degree of the interpolating polynomial is even (odd number of interpolating points), and the odd tabular point must be chosen arbitrarily, that with the smaller index is used.

AITKN is entered via the call statement:

```
CALL AITKN(X,Y,N,K,XB,YB,TEMP,IEX) .
```

The calling program must set X, Y, N, K, XB. These are not altered by AITKN. AITKN sets YB, IEX and alters TEMP.

X	: list of values of the independent variable in either increasing or decreasing order, a one-dimensional array
Y	: list of corresponding values of the dependent variable, a one-dimensional array
N	: the number of X,Y pairs
K	: the degree of the interpolating polynomial; $K \leq N-1$
XB	: the value of the independent variable to be used for interpolation
YB	: the interpolated value of the dependent variable
TEMP	: a one-dimensional array of at least 2(K+1) words used by AITKN for temporary storage
IEX	: either 1 or 0 according as extrapolation was or was not performed.

The user is cautioned that increasing the degree of the fitted polynomial does not necessarily increase the "accuracy" of the interpolation; indeed, this may introduce spurious oscillations into the interpolated values since the higher the degree of a polynomial, the more relative maxima and minima it can have. In particular, if the tabular data involves appreciable "noise", then use of a technique which "smooths" rather than one which "fits" (as does AITKN) may be more appropriate.

Subroutine AXSYUF (I1,I2,ANGL,FS,FI,SIGYB,SIGYE,P,RCG,FLAG)

AXYUF finds the ending stress, load, and center of load on a deformation element subject to axisymmetric, upset strain. A complete explanation of the technique used to find the above values is given in Appendix III.

The variables in the calling sequence are as follows:

(1)	I1	- The index to the left boundary of the deformation element (input).
(2)	I2	- The index to the right boundary of the deformation element (input).
(3)	ANGL	- The wedge angle, in radians, of the section (input).
(4)	FS	- The flow stress of the material; taken to be the material yield strength at the forging temperature (input).
(5)	FI	- +1 if material flow is left to right; -1 if material flow is right to left (input).
(6)	SIGYB	- The ending stress on the element (output).
(7)	SIGYE	- The beginning (known) stress on the element (input).
(8)	P	- The vertical load on the element (output).
(9)	RGG	- The center of loading on the element (output).
(10)	FLAG	- A logical variable used to indicate that load calculations are to be made (input).

Subroutine AXYSLD (FS,RB,RE,HB,HE,SIGYE,SIGYB,P)

AXYSLD is used by subroutine NERGY to find the load on an element in axisymmetric converging flow at a shear angle of 35 degrees. The details of the equations used are given in Appendix III and V.

Subroutine CENTER (X,Z,R,XC,ZC,ALPHA,SGN)

Given three points defining an angle and the radius which is to be fitted to the angle, CENTER determines the location of the center and the two points of tangency. The details of the procedures used are given in Appendix II. The parameters passed in the calling sequence are as follows:

(1) X,Z: One-dimensional arrays, of three elements each, defining the coordinate points of the angle (input).

(2) R: An array of three elements where the second element defines the radius to be fitted to the angle (input).

(3) XC,ZC: Three-dimensional arrays where the first elements are the coordinates of the center of the radius, and the second and third elements are the coordinates of the tangency points (output).

(4) ALPHA: The angle subtended by the arc to be fitted (output).

(5) SGN: Sign of the sense of rotation of the arc; positive if counter-clockwise, negative if clockwise (output).

Subroutine CLPTS (XI,ZI,N,R,XO,ZO,M,XT,ZT)

CLPTS generates cutter offset positions along X-Z sections when the system is in the CAM phase of operation. It is called from PRTSRF. Starting with N input coordinates (XI,ZI), CLPTS first generates an array which represents the range in X divided into M equally-spaced coordinates. That is, let

$$DX = XI_{max} - XI_{min})/(M - 1) \quad , \tag{VII-5}$$

then

$$XT_1 = XI_{min} \tag{VII-6}$$

$$XT_2 = XI_{min} + DX \tag{VII-7}$$

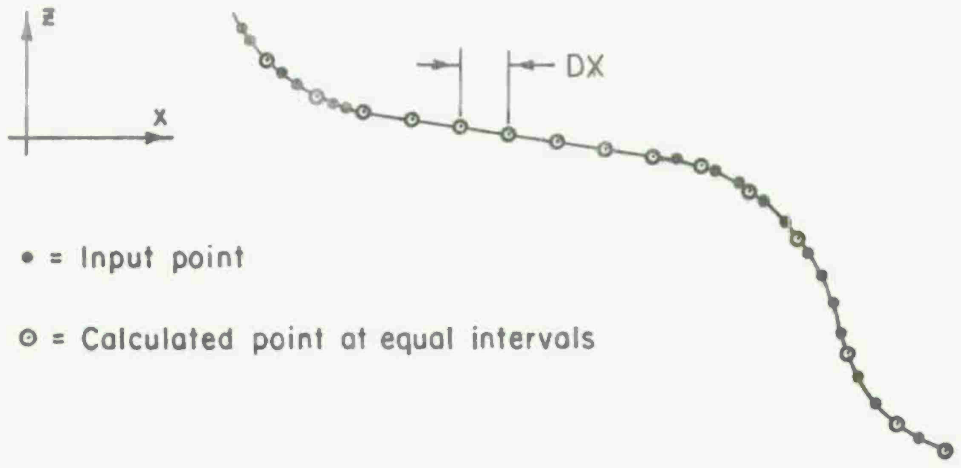
$$XT_3 = XI_{min} + 2DX \tag{VII-8}$$

$$XT_M = XI_{min} + (M - 1)\ DX = XI_{max}. \tag{VII-9}$$

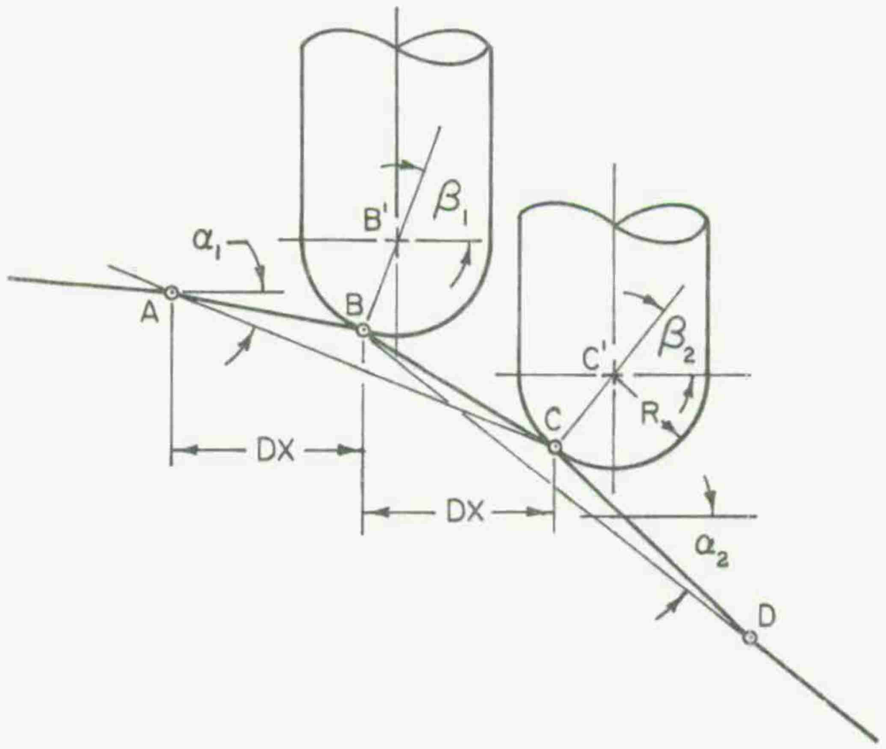
The Z coordinates, ZT, associated with each XT value are found using subroutine LINEAR. This set of points (XT,ZT) lie on the same surface as the input set of points (XI,ZI), but are uniformly spaced along the surface. It should be recalled that in this situation, the input points describe an interpolated surface and not an input polygon. Thus, depending on the number, M, of points calculated, array (XT,ZT) may be a very close linear approximation to the true circular arcs of the surface. The input and calculated points are shown in Figure VII-3A.

The calculated points are then used to find another new set of coordinates. This latter set is offset from the surface by the radius of the ball-end mill. This second set is the coordinates of the center of the cutter which will be used to machine the preform. The NC machine-tool positions the cutter using the center of the ball, rather than the point of contact of the ball and the desired surface. Therefore, the cutter position must be offset from the surface. This is done by taking each consecutive set of three points and finding the angle between the first and third points. Referring to Figure VII-3B, for Points A, B, C, angle α_1 is found as

$$\alpha_1 = \arctan\ ((Z_c - Z_A)/(X_c - X_A))\ . \tag{VII-10}$$



A. Generation of Uniformly Spaced Points



B. Generation of Cutter Offset

FIGURE VII-3. FUNCTIONS PERFORMED BY SUBROUTINE CLPTS

The angle of the normal to line AC then becomes

$$\beta_1 = \alpha_1 + \pi/2 \; . \qquad \text{(VII-11)}$$

β_1 is assumed to be the bisector of angle ABC. This is an approximation, but when the number of points is large and DX small relative to the overall surface, the error is acceptably small. The offset coordinates of the ball-end mill are found from β_1 as:

$$X(B') = X(B) + R\ (\cos \beta_1) \qquad \text{(VII-12)}$$

$$Z(B') = Z(B) + R\ (\sin \beta_1) \; . \qquad \text{(VII-13)}$$

The offset for the next point, C', is found in a similar manner, using Points B and D to find α_2 and β_2.

After the entire offset array is developed, DELETE is used to eliminate overlapping points. Overlapping points occur when a point has a greater X-axis value than the point immediately adjacent to it on the right (increasing X) side. Overlaps may occur when a surface changes from nearly vertical to nearly horizontal over a very small range, and when a large cutter is being used.

The array of points resulting from DELETE is again divided into a uniformly spaced (in X) array, using the procedures described above. LINEAR is then used a second time to find the Z-axis values associated with each of the uniformly spaced, offset X values.

The variables in the calling sequence are as follows:

XI, ZI:	Coordinate arrays for the input data
N:	The number of input points
R:	The radius of the ball-end mill to be used to machine the surface
XO, ZO:	Coordinate arrays for the output data
M:	The number of points to be generated in the output arrays
XT, ZT:	Temporary work area arrays of length M.

Subroutine DBLFST

This subroutine finds the neutral surface and maximum stress for a section with flash on both sides, be it in plane or axisymmetric strain. When called, it first initializes indices for deformation elements on the left and right sides, equates the initial stress on the extreme left and right flash boundaries to the material flow stress, and clears summary variables for the load and moment. A logical flag is also set true to indicate that load calculations are to be made.

It then determines whether a left or right-hand element has the lower stress. The routine evaluates the next element for the side which has the lower stress. Elements are alternately picked as necessary from the left and right sides in an attempt to keep the stress on each side in balance. The shear distribution in Figure VII-4 illustrates how the procedure alternates sides in an attempt to keep the stresses balanced. Working from either side towards the middle, the point where the elements meet will be determined, as described below. Having branched according to which side is to be evaluated, DBLFST then updates the indices which point to the coordinates of the elements. The beginning stress on the element is equated to the ending stress on the element on the same side which is adjacent to the current element. Either subroutine PLSTRS (plane-strain sections), or AXSYUF (Axisymmetric strain) is then called to find the stress at the end of the element, along with the load and center of load on the element. Upon returning from the appropriate subroutine, a test is made to determine if the elements from either side have met somewhere in the middle. If they have not, the running sums for load and moment are updated. The routine then loops back and determines which element to use next.

Where the elements meet is determined by comparing the indices to the boundaries on each side. The elements have met when the index to the right-hand boundary of a left-hand element (IL2) equals the index to the left-hand boundary of a right-hand element (IR1), as seen in Figure VII-5 In Figure VII-4, this occurs at boundary Index 4. The next step is to determine if it is necessary to interpolate within an element to find the neutral surface. The neutral surface is the point where the stress from one side is equal to the stress from the other side. Interpolation is not used when the

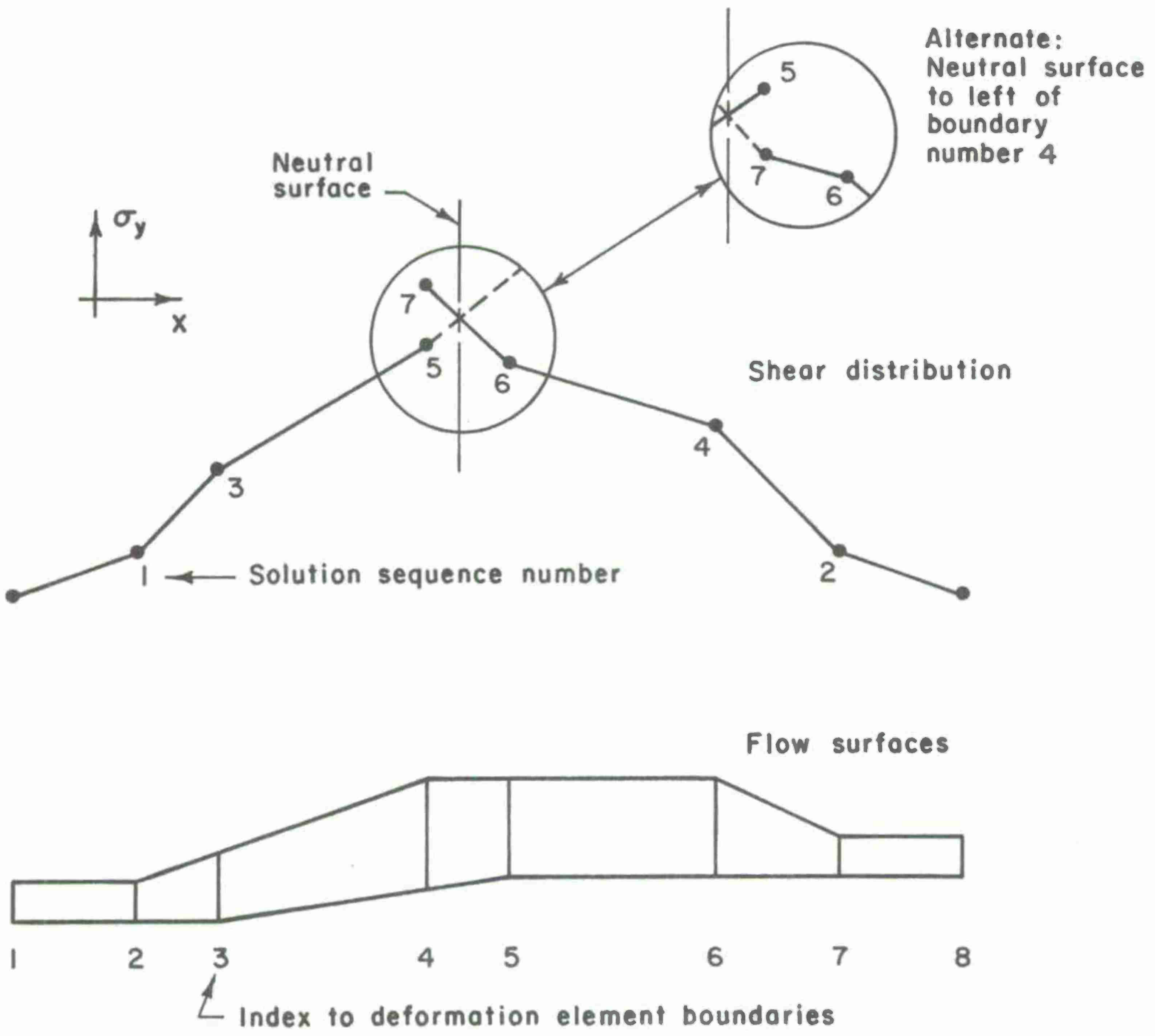


FIGURE VII-4. LOCATION OF BOUNDS OF NEUTRAL SURFACE

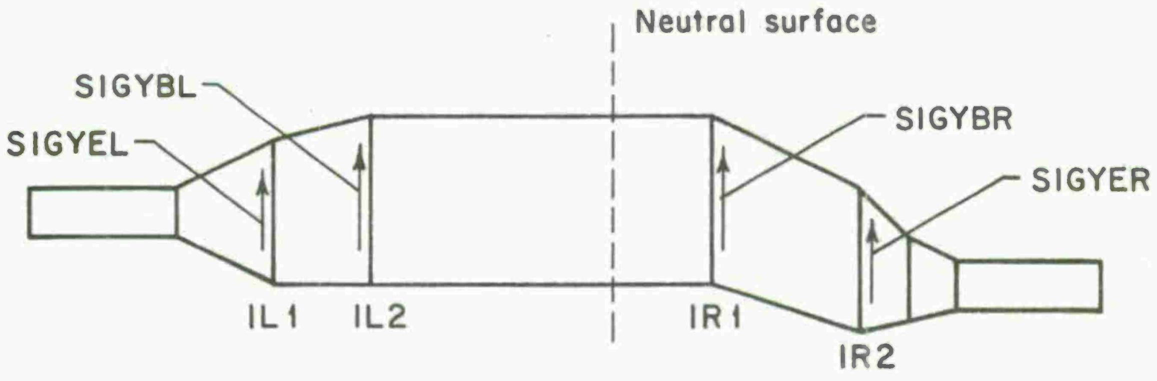


FIGURE VII-5. LABELING CONVENTIONS USED IN SUBROUTINE DBLFST

difference between the two ending stresses is less than 1 percent of the average of the two. When this condition holds, the neutral surface is taken to be the coordinate of the deformation element where the two stresses met.

If interpolation is needed, DBLFST determines the boundaries between which the neutral surface lies. If the ending stress from the left is less than the ending stress from the right (Figure VII-4), the neutral surface lies to the right of the boundary where the elements met, (i.e., between boundaries 4 and 5, Figure VII-4). When the ending stress from the left is greater than the ending stress from the right, the neutral surface lies to the left of the boundary where the elements met (see inset, Figure VII-4).

When the boundaries have been determined, the beginning stresses on the element from each side are found. The actual solution for the neutral surface is an iterative one. Referring to Figure VII-6, a first guess as to the position of the neutral surface is made by assuming it lies half way between the two boundaries. This value is then used to determine the other coordinates of the new element and the stresses from either side are then found. If the resulting ending stresses are not equal (or nearly so), a new guess for the X coordinate of the neutral surface is made. This is done by halving the previous difference, and adding or subtracting it to the previous guess. While cycling through this process, the flag is set false to avoid the computations necessary to find the load and center of load.

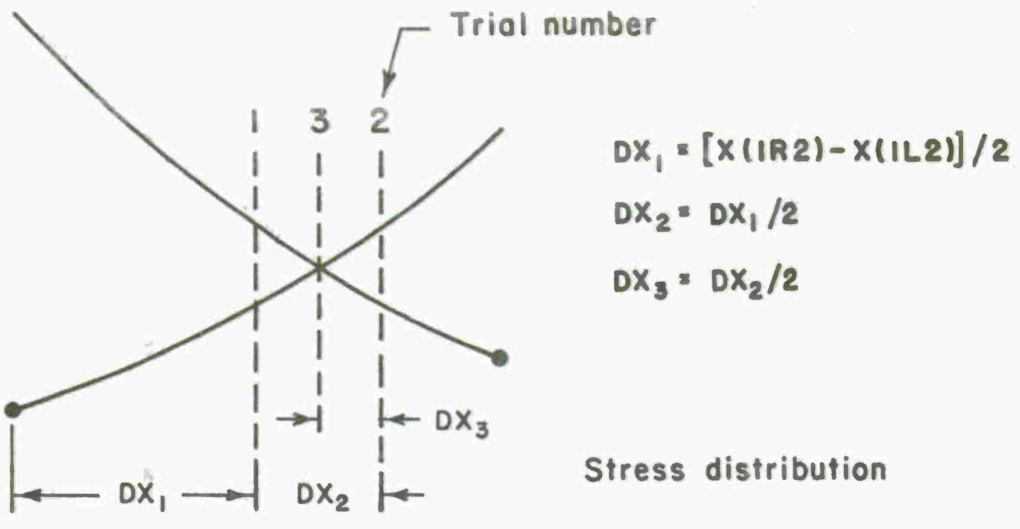
Once the neutral surface has been found, its value and that for the maximum stress is saved. The flag is then set true and the stress routines called to calculate the loads on the two new elements. After each call, the sums for load and moment are updated. Finally, the center of load is found by dividing the sum of the moments by the sum of the loads.

Data is passed to and from this routine via Blank Common, and Labeled Common SECTN, SYPLOT, and SYSPR.

Significant variables within DBLFST include:

(1) IL1: The index to the left-hand boundary of a deformation element to the left of the neutral surface.

(2) IL2: The index to the right-hand boundary of a deformation element to the left of the neutral surface. The coordinate represented by this index may be the neutral surface.



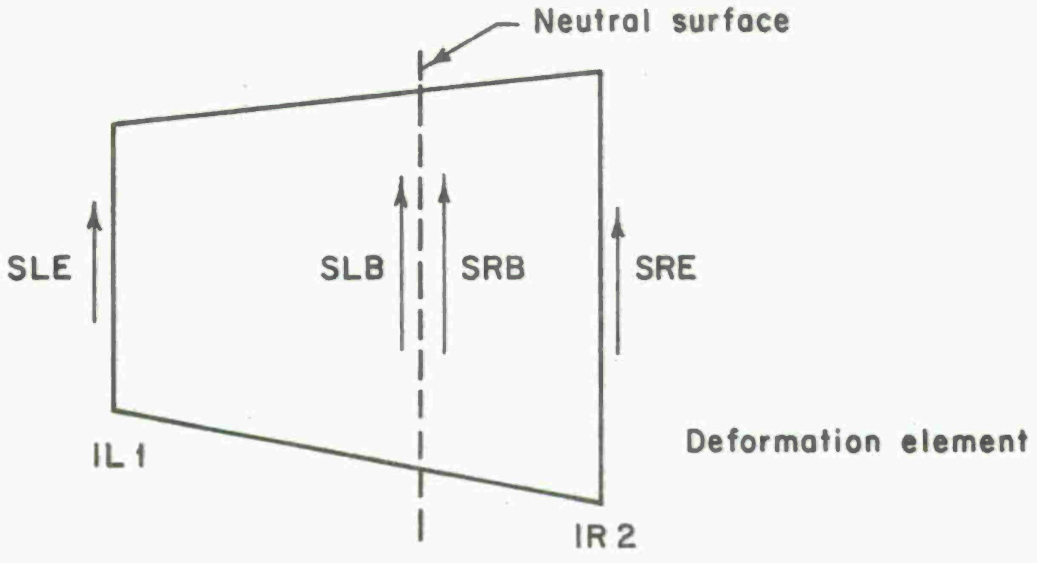


FIGURE VII-6. FINDING NEUTRAL SURFACE BY ITERATIVE "BRACKET AND HALVE" TECHNIQUE

(3) IR1: The index to the left-hand boundary of a deformation element to the right of the neutral surface. This may be the index to the neutral surface.

(4) IR2: The index to the right-hand boundary of a deformation element to the right of the neutral surface.

(5) SIGYEL, SLE: The stress on the left boundary of a deformation element to the left of the neutral surface. This is the known stress for a left-hand element.

(6) SIGYBL, SLB: The stress on the right boundary of a left-hand deformation element. This is the unknown stress to be solved for.

(7) SIGYER, SRE: The stress on the right boundary of a right-hand deformation element. This is the known stress for a right-hand element.

(8) SIGYBR, SRE: The stress on the left boundary of a right-hand deformation element. This is the unknown stress to be solved for.

Subroutine DECUSP (X,Z,R,N,M,IFLAG)

This subroutine tests an N-sided polygon defined by coordinate arrays X and Z, each with an associated radius R, to determine if cusps will form where one radius is to blend smoothly with another. A cusp condition is illustrated in Figure VII-7 by Points AEFGHD. If a cusp condition is found, both radii are reduced by an equal percentage.

DECUSP first determines if the data represents a closed figure, or only an upper or lower surface. This is done by testing parameters M as follows:

If $M = 1$, closed figure

If $M \neq 1$, upper or lower surface.

If the data represent a closed figure, the first point is copied as the last point so that all radii are evaluated.

Referring to Figure VII-7, each set of four consecutive points on the polygon are evaluated. For the two radii associated with the four points, the distances from the polygon points to the tangent points with their associated radii are found using subroutine YOFFST. For points ABC and radius R1, this distance is L1; for Points BCD and Radius R2, the distance is L2. The distance

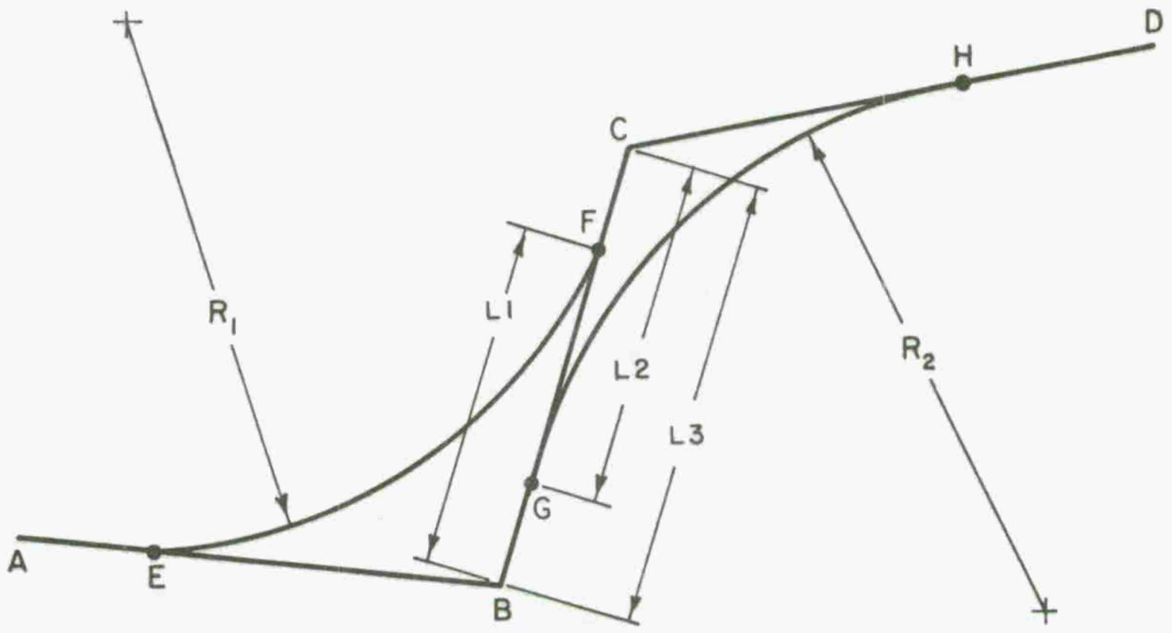


FIGURE VII-7. PROCEDURE TO ELIMINATE CUSPS

H3, between the second and third points of the four being evaluated, is also found as

$$L3 = [(X_B - X_C)^2 + (Z_B - Z_C)^2]^{1/2} \quad . \qquad \text{(VII-14)}$$

A cusp exists when

$$L3 < L1 + L2 \quad , \qquad \text{(VII-15)}$$

$$\text{or} \quad L3/(L1 + L2) < 1.0 \quad . \qquad \text{(VII-16)}$$

If no cusp exists, the index is incremented and the next set of four points is evaluated.

If a cusp exists, the two radii are reduced by the ratio L3/(L1 + L2). The program then loops back and makes another test for a cusp using the same four points and the two modified radii. It is believed that the radii, after being modified by the above ratio, will not cusp. However, this has not been analytically proven, and the re-test ensures that eventually the cusp is eliminated.

The variable IFLAG determines if the radii are to be modified after a cusp condition is found. It is used as follows:

If IFLAG ≠ ϕ, determine if cusps exist but do not remove.

If IFLAG = ϕ, find and remove all cusps.

The variables used in the calling sequence are as follows:

X,Z:	Arrays of coordinate data defining a polygon
R:	An array defining the radii associated with each point on the polygon
N:	The number of points on the polygon
M:	Used to indicate whether polygon is a surface or a closed figure
IFLAG:	Used to indicate whether cusps are to be removed or not.

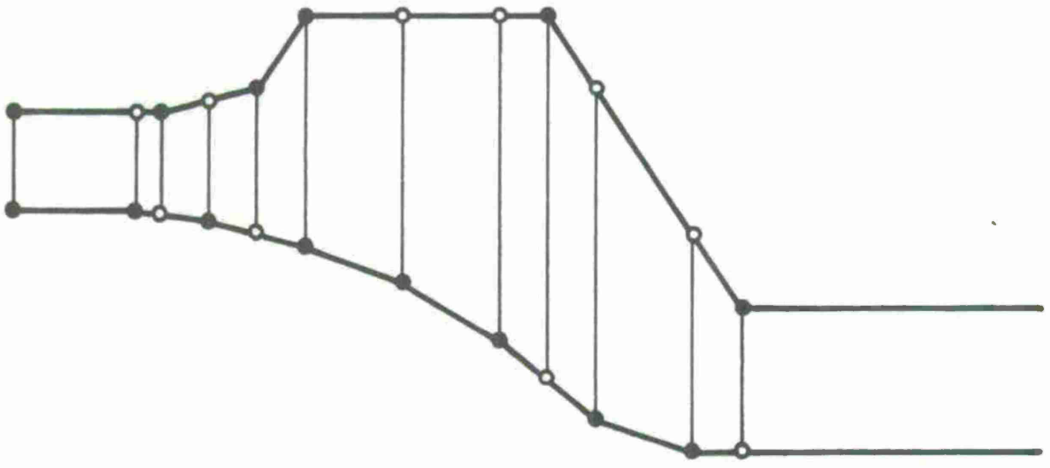
In addition to the above parameters, data is also passed via Named Common block SYSPR.

Subroutine DEFREL

This subroutine generates the coordinates of trapezoidal deformation elements. These are formed by a series of vertical lines bounded by the upper and lower flow surfaces. In addition, the routine determines the proper friction factor for the upper and lower surface of each trapezoid. The routine prevents the generation of trapezoids of less than a specified minimum width by eliminating such points. Each deformation trapezoid boundary is defined by one X value and two Y values.

DEFREL starts by defining the left boundary as equal to the X coordinate of the first upper surface point with the Y coordinates equal to the first Y values of the upper and lower surfaces (see Figure VII-8). If the part is axisymmetric, single flash, the first output coordinates are equated to the coordinates of the second upper and lower surface points. This is because for axisymmetric, single flash sections, the second point for each die surface lies directly above, or below, the first point. An index is then initialized for the number of points in the output array (the number of deformation elements). Pointers to the next upper or lower surface point to test are also initialized.

The next point to add to the output file is then found by comparing the X value of the next upper surface point to the X value of the next lower surface point. The lesser of the two is selected. The X value of this point is compared to the X value of the point most recently added to the output file. If the two are further apart than a specified minimum, the X and Y values for the surface point are placed in the output file. The Y value for the opposite surface is found by a call to the function XYINTR. This finds the needed Y value by interpolating between the two points on the opposite surface which bound the latest point. If the new point is closer than the minimum to the previous point, a test is made to determine if the latest point is a shear boundary. If it is not a shear boundary, the latest point is ignored. If it is a shear boundary, the previous point saved is replaced by the current values. After each surface point is evaluated, the counter for that surface is incremented. When both counters are equal to the number of points on each respective surface, the process is completed. Otherwise, the routine loops back and compares the X values of the next two surface points. When the testing process is completed, the X and two Y values for



● = Flow surface point

o = Point found by interpolating opposite surface

FIGURE VII-8. GENERATION OF TRAPEZOIDAL DEFORMATION ELEMENTS

the right-hand flash are added to the file. This overall process results in the generation of three arrays. One array is for the X values, with the other two arrays being for the upper and lower Y values for each X.

Arrays for the friction factors acting on the surfaces to the right of each deformation element boundary are found as follows. The output array is scanned from left to right, testing each value to see if a shear boundary has been passed. If a shear boundary has not been passed, the friction factor for the current surface is the same as it was for the previous surface. If a shear boundary has been passed, the friction factor is set to the opposite of what it was previously. This process is done separately for the upper and lower surfaces of each deformation element. This process is done for one less than the total number of deformation element boundaries, since no surface exists beyond the right-hand flash point.

Data for this subroutine is passed via Blank Common, and Labeled Common STRSPT and SECTN.

Subroutine DELETE (X,Y,N)

This subroutine eliminates redundant, overlapping, or extremely close points from the display file. It deletes points from the arrays for the following reasons:

(1) If X(I+1) < X(I). This ensures the X values of the result are constantly increasing.

(2) If the absolute distance between adjacent points is less than 0.002 inch. This eliminates one of the two points which are equal, or nearly so.

The variables in the calling sequence are:

(1) X,Y: One-dimensional arrays of N elements each (input and output).
N: The number of elements in the arrays (input and output).

Subroutine DIESRF

DIESRF changes the definition of a section from a series of coordinates describing a polygon in clockwise order, to two sets of coordinates, one each defining the upper and lower die surfaces. While doing this, it also adds the coordinates which define the flash. The coordinates of the first point on the upper die surface are derived from the left-hand parting line coordinates and the flash dimensions as shown by Equations (VII-17) and (VII-18), and Figure VII-9.

$$XT(1) = X(1) - \text{Flash Width} \tag{VII-17}$$
$$YT(1) = Y(1) - \text{Flash Thickness}/2 \quad . \tag{VII-18}$$

The Y coordinate of the second flash point is equal to that of the first, since the flash lands are always assumed to be horizontal. The X coordinate of the second flash point is found by calling function XYINTR. The balance of the section points up to, but not including the right-hand parting line point, are then transferred to the upper surface file. The right-hand flash points are found in a process similar to that used to find the left-hand flash points. The bottom surface coordinates are found in a similar manner. After both surfaces are calculated, subroutine DECUSP is used to eliminate any cusp conditions between adjacent radii.

Data is passed to this routine via Named Common Block SECTN and FINISH.

Subroutine ERRPRT

ERRPRT is used to print error messages regarding problems encountered in subroutine RDIOF4. The messages are output to both the CRT and the printer disk file. After writing to the CRT, the program pauses until a carriage return is typed.

Once called, the program uses the variable IERROR as the argument in a directed GO TO. This causes the program to branch to the appropriate WRITE statement. If IERROR equals 1, no error condition exists. ERRPRT will not be called when no error exists.

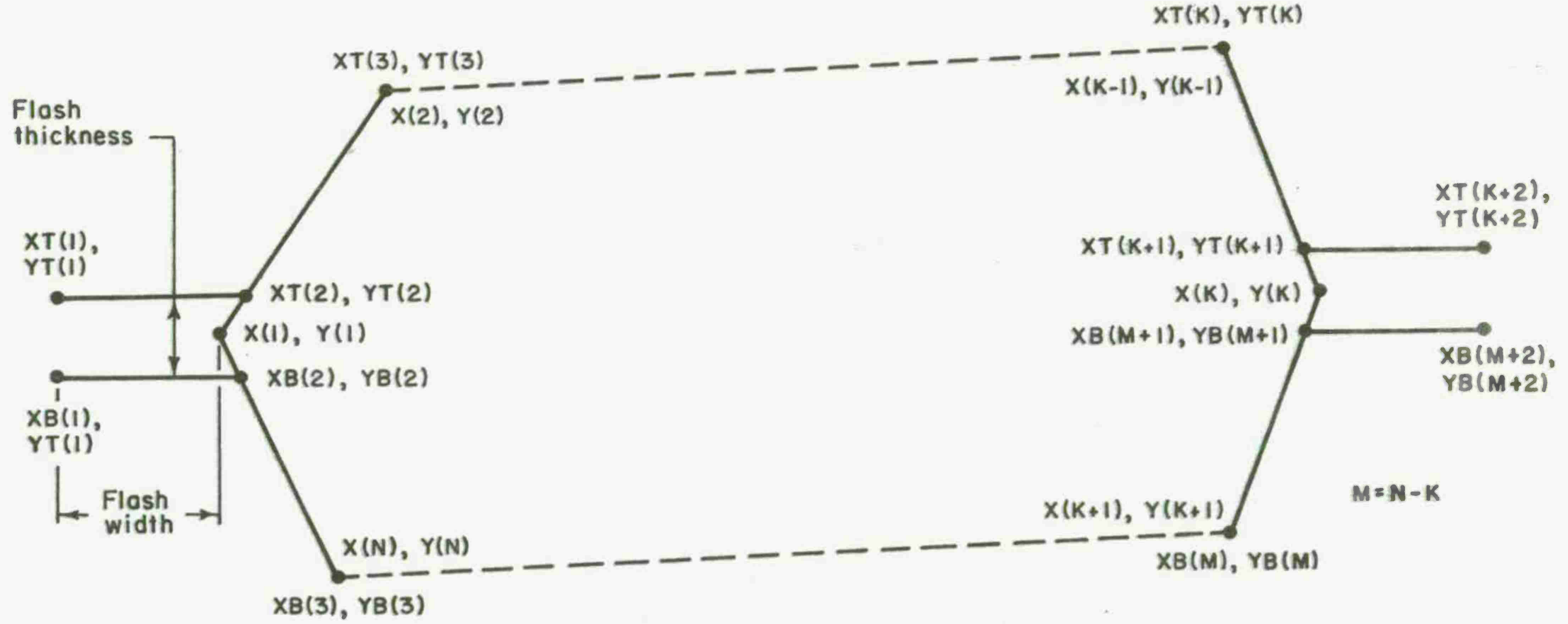


FIGURE VII-9. COORDINATES DEFINING UPPER AND LOWER DIE SURFACES, DERIVED FROM SECTION POLYGON AND FLASH DIMENSIONS

The following error messages are currently implemented:

Error Code	Message or Meaning
2.	No section header - end of data assumed.
3.	Too many coordinate points. Will attempt to process the next section.
4.	Too few coordinate points. Will attempt to process the next section.
5.	An end-of-file designator was encountered. The program will terminate.
6.	Too many data points on upper surface. Will attempt to process the next section.

Data is passed to ERRPRT via the Labeled Common Blocks SYSPR and SECTN.

Subroutine FILTER (XO,ZO,NO,XS,ZS,NS,XF,ZF,NFF)

The primary function of this subroutine is to reduce the number of points defining a flow surface to 50 or less. It also eliminates points which are closer than a specific distance in X. The routine first places the first two points of the unfiltered flow surface in the file of filtered points. These points either define the left-hand flash, or the axis of symmetry, and are always included in the output. The upper bound for the number of points to allow in the output file is set to one less than the total allowable. This is because the last point is a flash boundary and is always used.

The minimum distance between points in terms of X in the output file is then set. This is initially set to 0.010 inch, and is increased by 0.010 as many times as is necessary so that the output file contains 50 or less values. Indices are next initialized for the count of points in the output array, the pointer to the present test point, and a pointer to the next input array point to be tested after a point is put in the output array. The upper bound for points to test in the input array is set to one less than the total count of points in the input array. This is because the last point is the extreme rightmost flash coordinate and is always included in the filtered output. An index to the shear boundaries is initialized to one. However, if the section is in axisymmetric strain with single flash and has a shear boundary on the axis of symmetry, this index is set to two. Shear boundaries are always included in the file of filtered points, regardless of whether or not they lie closer to the previous point than the allowable minimum. When the above conditions exist, the first shear boundary will have already been placed in the array of filtered points by virtue of it being the second point in the file of unfiltered points.

The first test point index and the current test point index are then both incremented by one, followed by a test of the current test point index to see if all points have been tested. When they have, the rightmost flash point is added to the output array, and the routine exited. When additional points remain to test, the distance in X, between the current test point in the input array and the last point placed in the output array is found. If the distance is more than the current minimum, or if the test point is a shear boundary, the routine branches. If not, the current test point index is incremented by one and the next point evaluated.

Referring to Figure VII-10, assume Point A has been placed in the output array. Point B, the next point, is the Ith point in the input array. The current test point index and the pointer to the next input array point are both set equal to I, and the horizontal distance, D, between B and A are found. Since this is less than the current minimum, R, point B is not accepted. The current test point index is incremented so it points to C. The distance between C and A is greater than R, but the current test point index and the pointer to the next input file point are not equal, so B' is placed in the output file. The coordinates of B' are:

$$X(B') = X(A) + R \qquad \text{(VII-19)}$$

$$Y(B') = X(B) + (Y(C) - (Y(B))\,\frac{(X(B') - X(B))}{(X(C) - X(B))}\,. \qquad \text{(VII-20)}$$

The index to the current test point and the pointer to the first test point are both then set equal to the index to C. Since the distance between C and B' is greater than R, and the current test point index and the first test point index are equal, C is added to the output file. This results in the filtered surface being defined by AB'C with no points closer together, in X, than R.

If B had been a shear boundary, it would have been placed directly in the output file, regardless that it was less than R away from the previous point. This would have resulted in the filtered surface being defined by ABC.

When all input points, except the last one, have been evaluated in a similar manner, and the result is 50 or less points, the right-hand flash point is appended to the output file, and a return made to MERGE.

Calling sequence arguments:

XO,YO:	Arrays defining the coordinates of the merged, but unfiltered flow surface (input).
NO :	The number of points defining the input surface (input)
XS,YS:	Arrays defining the coordinates of shear boundaries (input)
NS :	The number of shear boundaries plus one (input)
XF,ZR:	Arrays defining the coordinates of the merged and filtered flow surface (output)
NFF :	As input, the maximum allowable number of points defining the filtered flow surface. As output, it is the actual count of points defining the filtered flow surface.

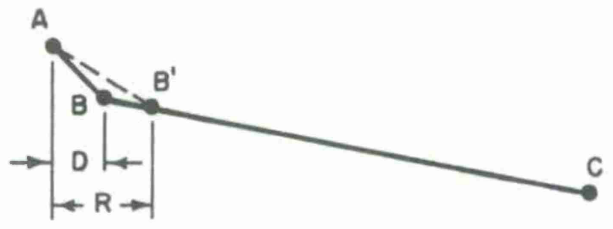


FIGURE VII-10. FILTERING TECHNIQUES TO ELIMINATE UNNEEDED POINTS FROM FLOW SURFACE

Subroutine FITARC (X1, Z1,X2,Z2,X3,Z3,RD,XP,ZP,NP,RES,LIM,IFLAG)

FITARC calculates the coordinates of line segments which can represent a radius to a given resolution. A complete description of the technique used is contained in Appendix II.

The variables in the calling sequence are as follows:

(1) (X1,Z1), (X2,Z2), (X3,Z3): three points defining two intersecting lines (input).

(2) RD: the radius at the intersection to be simulated by straight-line segments (input).

(3) XP and ZP: X and Y coordinates of the newly defined line segments (output).

(4) NP: the total number of points calculated for a section (output).

(5) RES: the arc distance between consecutive points on the radius (input).

(6) LIM: the maximum number of points which can be calculated for a section (input).

(7) IFLAG: a flag indicating that NP equals LIM (output).

Subroutine FLWSRF

This subroutine controls the logic flow which generates the material flow surfaces. When these surfaces, across which material will flow when the forging is made, are defined, the stress on the surfaces can then be determined.

When called, FLWSRF first asks the designer to enter the friction factor between the material and the die. A default value of 0.25 is used if no value is entered. Instructions as to how shear boundaries are to be indicated are then typed and the upper and lower die surfaces are made light pen sensitive. Metal flow in shear will generally occur across the base of any rib in a forged part. Once the cavity in the die, which forms the rib, is filled, metal no longer flows into the cavity. Instead, as the dies continue to close, the material shears near the base of the rib. It is necessary for the designer to indicate the points on the die surface where metal flow in friction changes to metal flow in shear. Each cavity where shear is assumed to take place must be marked with two points.

FLWSRF uses subroutine GETPTS to obtain the shear boundary points indicated by the designer with the light pen. If more than 10 points are indicated on either surface, an error message is generated and GETPTS is called again. Subroutine GETPTS is followed by subroutine FORCE which ensures the coordinates of the shear boundaries lie absolutely on the die surface. Subroutine ADDPTS is then used to add two additional points to each shear surface. These extra points are generated for each pair of shear boundaries.

Subroutine MEGRE is used to generate the top and bottom flow surfaces. The flow surface is composed of the original die surface where material flow in friction occurs and the shear surfaces defined by the shear boundaries and the intermediate shear surface points. A maximum of 50 points each is used to define the upper and lower flow surfaces, so MERGE uses FILTER to reduce the number of coordinates to this quantity. Finally, subroutine DEFREL is used to form the deformation elements from the two surface definitions. These deformation elements are the entities used to determine the stress on the section by means of "slab method" analysis. After the deformation elements have been found, the flow surfaces are displayed with the shear boundaries marked with crosses. Figure I-6 shows the flow surfaces for a typical section.

Figure VII-11a shows a typical die cavity (part rib) with the shear boundaries as obtained from GETPTS and FORCE, and the intermediate shear surface points from ADDPTS. Figure VII-11b shows the resultant flow surface after processing by MERGE. Figure VII-11c is the flow surface after FILTER has been used to reduce the member of points on the surface to 50 or less. Figure VII-8 illustrates the formation, from the flow surfaces, of the deformation elements used for stress analysis.

Data for this subroutine is obtained from Blank Common and Labeled Common blocks STRSPT, SYSPR, SECTN, DISPLA, and DEFEL.

Subroutine FORCE (MARK,X,Y,XO,YO,N)

FORCE finds the point on a surface which is the closest to a given point. It does this by evaluating the triangles formed by each pair of adjacent surface points and the given point. From these triangles, FORCE uses either the least distance between the given point and a surface point, or the normal distance from the given point to the surface. A complete

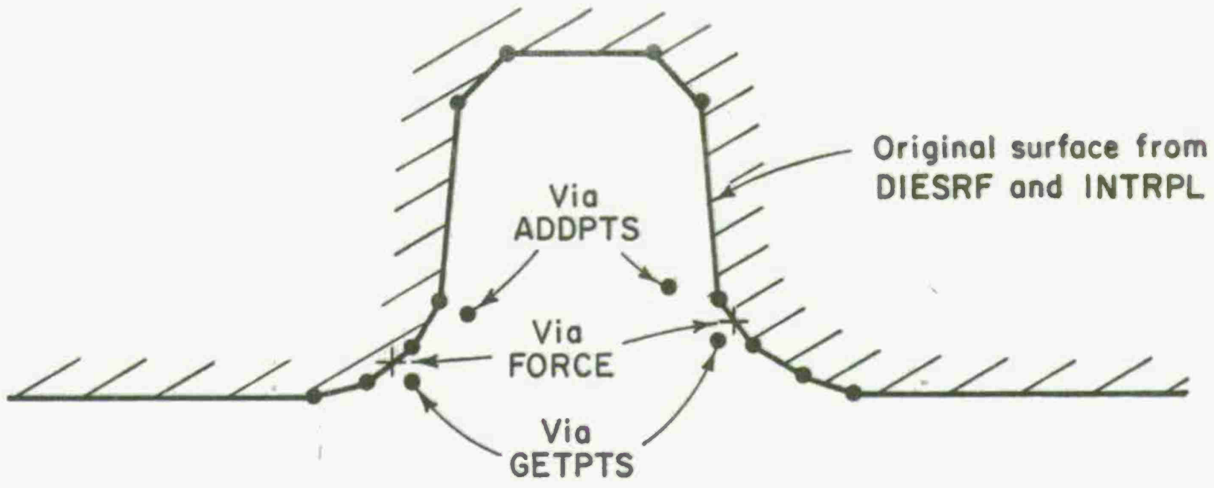


FIGURE VII-11a

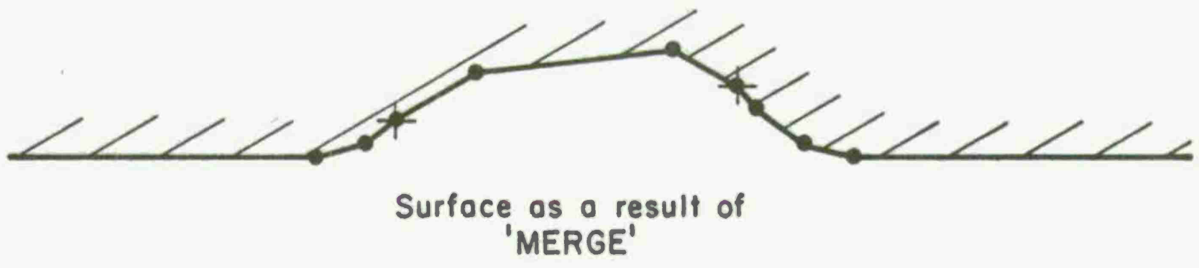


FIGURE VII-11b

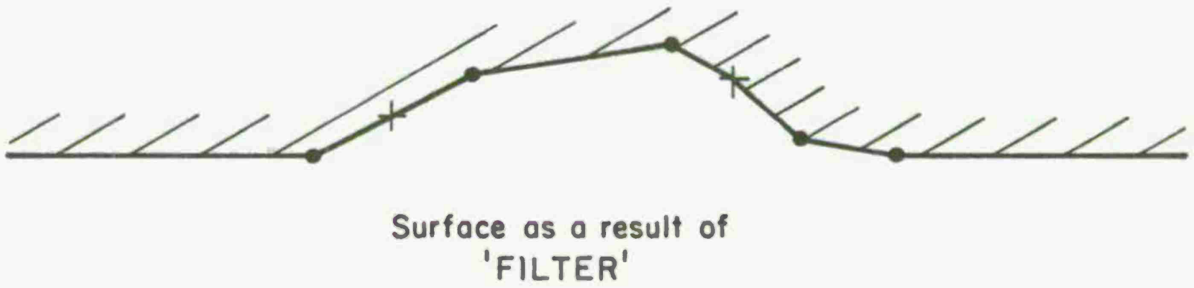


FIGURE VII-11c

description of the technique used is given in Appendix II. Since the given points are entered in left to right order in GETPTS, it is not necessary to search the entire surface for each given point. Instead, the rightmost point of the triangle which contains the solution for a point becomes the leftmost point of the first triangle to evaluate for the succeeding point.

Arguments in the calling sequence:

(1) MARK: the index to the surface point at which the searching process is started (input).

(2) X,Y: as input, the coordinates of the given point. As output, the coordinates of the point on the surface closest to the given point.

(3) XO,YO: one-dimensional arrays containing the coordinates of the surface (input).

(4) N: the number of coordinates defining the surface (input).

Subroutine GETHIT (K,J,L,J1,J2)

Subroutine GETHIT obtains light-pen (L.P.) hits on sensitized display items, and returns the tag numbers for each item hit. The X and Y coordinates of the last light-pen hit are also returned.

When first called, GETHIT restores the "ACCEPT" text as an L.P. sensitive item. It then starts a Do-loop to get L.P. hits on K items. K number of items must be picked in order for this routine to terminate. After starting the loop, a dummy, nondisplayed point is created. This will be replaced when an item is actually hit, and a cross (+) is drawn to indicate the position of the hit. A call to LGTPEN is then issued which returns the item number of the entity hit with the L.P., and the coordinates of the hit. If the first hit is on the "ACCEPT" text, the program loops and looks for another hit, since an item cannot be accepted until it is identified. If the hit was on a display item, the dummy point is deleted and replaced with the coordinates of a cross (+) to indicate where the hit was made. The item tag returned from the valid L.P. hit is saved in array K, and the coordinates of the hit saved in J1 and J2.

Another L.P. hit is then looked for. If the second hit is on the "ACCEPT" text, the item previously hit is desensitized so it will no longer respond to L.P. hits. The value "L" is then subtracted from the tag value previously saved. This allows the indices returned to represent the actual array indices for the point, rather than the display item index. The Do-loop then continues or terminates, as appropriate.

If the second hit was not on "ACCEPT", but rather on another L.P. sensitive display item, the program loops back. The display item for the cross at the previous position is deleted and replaced with a cross at the current location. Once the Do-loop is satisfied, the "ACCEPT" text is blanked before the subroutine returns control to the calling program.

Data is passed via Labeled Common block DISPLY, and the following parameters in the calling sequence:

(1) K: the name of the array used to store the modified display item tag values (input and output).

(2) J: the number of L.P. hits to be obtained (input)

(3) L: the offset between display items and the associated element of the coordinate arrays (input).

(4) J1 and J2: the X and Z coordinates of the last point hit with the L.P. (output).

Subroutine GETPTS

GETPTS allows the designer to indicate the location of cavities on a die surface. Two points are required to define a single cavity. Metal flow in shear is assumed to occur between the two points, while metal flow in friction is assumed to occur in all other sections of the surface.

When first started, GETPTS initializes an error flag, and counters for the number of points indicated on the upper and lower surfaces. The extreme horizontal values are then found for each surface. The right extreme value for each surface is the inside, right flash point. The left extreme value is either the inside, left flash point, for double flash sections, or the axis of symmetry for single flash sections. The left extreme value is actually located slightly to the left of the point just described. This is because, for single flash parts, a valid hit on the axis of symmetry could be rejected when the hit is scaled back to user dimensions. This is due to truncation errors when the figure is originally scaled for display.

The "END" and "ACCEPT" texts are then restored to view, the upper and lower part surfaces made light-pen sensitive, and a dummy, nondisplayed point is entered into the display file. The program then waits for the designer to indicate one of the light-pen sensitive items (two pieces of text and two surfaces). When a hit is received, it is first tested to see if "END" was indicated. If it was, the subroutine termination logic is executed as described below. If the hit was on a surface, the last item in the display file is deleted, and replaced by a cross (+) at the position of the hit. It is this procedure that allows the designer to move the light pen along a surface and have the cross follow the pen.

If "ACCEPT" was indicated and a surface point was indicated on the previous hit, the coordinates of the previous hit are scaled back to user dimensions. A branch is then made according to whether the upper or lower surface was the item previously hit. A test is made to determine if less than 10 points have been indicated for the surface. If 10 or more points exist for the surface, the error code is set and a return is made to FLWSRF. If less than 10 points exist, the latest point is compared to the extreme values for the surface. If the point lies between the two extremes, it is saved in an array, and the counter is incremented. If the point did not lie between the extremes, the point is not saved. Instead, it is deleted from the display file. In either case, a new dummy item is created and the process is repeated.

When the "END" text is indicated, the count of points for each surface is checked to see that points were entered in pairs for each surface. If there is an odd number of points on either surface, the program loops back, creates a new dummy item, and waits for a light-pen hit. If the points were entered in pairs for both surfaces, a last dummy point is artificially generated and added to the arrays of points indicated. This is used when determining the flow coefficient for each segment of the flow surfaces in subroutine DEFREL. Finally, the "END" and "ACCEPT" text strings are blanked, and the two surfaces made non light-pen sensitive.

Data for this subroutine is passed by means of Blank Common and Labeled Common blocks DEFEL, DISPLA, SECTN, and STRSPT.

Subroutine GETRID

Subroutine GETRID is used by PFPREP at the termination of the first stage of preform design, to eliminate unneeded points from the preform polygon. It eliminates points from the preform arrays which have radius values equal to 10^{30}. Values of this magnitude are assigned to temporary points when using the Add Point function, or to points picked when using the Delete function. GETRID operates by using a Do-loop to search the preform arrays and adjusting the index whenever a large value is found for a radius.

Data for this subroutine is passed via Labeled Common block DEFEL.

Module GT40 and GT40TC

These modules consist of a number of FORTRAN callable subroutines which allow the user to create graphic images and text on the GT40 display system CRT. Of the following calls, all but TRACK is in module GT40. Thus, if the tracking cross routine is not needed, GT40TC does not need to be included in the command string to the linker. Both modules are written in Macro assembly code.

GT40 Graphics Subroutine Package FORTRAN Call Summary

All coordinates are in screen units. X axis has a range of Ø - 1Ø24, y axis Ø - 768.

CALL BLANKF(DFILE)

To blank a display file.

CALL BLANKI(TAG)

Blanks the item identified by its tag.

CALL CHNGIT(TAG,INT,LPNS,BLNK,TYPE,ITALCS)

Changes the attributes of an item identified by its tag. All attributes are optional, and may be skipped by inserting a blank between corresponding commas.

<u>CALL DSPLYG(DFILE,L,ITEMS)</u>

This call establishes a buffer area for the graphics file items to be created by subsequent calls to LINESG, POINTG, etc.

DFILE:	Display file area; a one-dimensional array in the calling program.
L:	Length of the array.
ITEMS:	Number of items one expects to store in this Display File. Each call to LINESG, POINTG, etc., creates one item.

<u>CALL EXITG</u>

To terminate Graphics Mode.

<u>CALL LEGNDG(TAG,IX,IY,N,CHAR,INT,LPNS,BLNK,ITALCS)</u>

Displays a character string starting at IX,IY. The arguments INT, LPNS,BLNK,ITALCS are optional.

N:	Number of characters to be displayed.
CHAR:	An array containing the characters
INT:	Intensity
LPNS:	Light pen sensitivity
BLNK:	Blinking status
ITALCS:	Italics font or normal.

<u>CALL LGTPEN(TAG,IX,IY)</u>

To get the tag and coordinates of the first light pen hit after the call.

<u>CALL LINESG(TAG,N,IX,IY,INT,LPNS,BLNK,TYPE)</u>

Displays a series of vectors connecting the input coordinates. The arguments INT,LPNS,BLNK and TYPE are optional.

TAG:	Is the identifier assigned to the item created by the call to LINESG.
N:	Number of coordinate pairs.
IX,IY:	Coordinate pairs in object space units.
INT:	Intensity at which the item would be displayed Ø = dimmest, 7 = brightest.
LPNS:	Light pen sensitivity Ø = not sensitive, 1 = sensitive.
BLNK:	Blinking status, Ø = steady, 1 = blinking.

TYPE: Line type; Ø = solid
1 = long dash
2 = short dash
3 = dot dash.

CALL LNKDFH

To link to the Display File Handler (DFH). DFH would be used if GM is not in use. Thus, either LNKGM or LNKDFH should be called.

CALL LNKGM

To link to the Graphics Monitor (GM). Graphics Monitor is in use, if GTON or GTDIAL was executed.

CALL NEXTAG(ITAG)

The next available tag will be returned in ITAG.

CALL POINTG(TAG,N,IX,IY,INT,LPNS,BLNK)

Displays a series of points. The arguments INT,LPNS,BLNK are optional. Any of them may be omitted by inserting a blank in between the corresponding commas.

TAG:	The tag assigned to the item created by this call.
N:	Number of points to be displayed.
IX,IY:	Arrays containing the coordinates.
INT:	Intensity Ø - 7.
LPNS:	Light pen sensitivity.
BLNK:	Blinking status.

CALL PRNTCH(IC)

Print a character on the screen and/or the Decwriter.

CALL READCH(IC)

Get a character from the keyboard. If no character is available, IC = Ø.

CALL REMOVF(DFILE)

To purge a display file.

CALL REMOVI(TAG)

Removes all items with tags equal to and greater than TAG from the display file, reclaiming memory space. If TAG is greater than any existing item identifier, message "UNASSIGNED TAG" is returned.

CALL RESTRF(DFILE)

To restore a blanked display file.

<u>CALL RESTRI(TAG)</u>

Restores a blanked item.

<u>CALL SCROLG(NLINES,IYTOP)</u>

To adjust scroller parameters. Graphics Monitor must be in use.

NLINES:	Number of lines to be displayed.
IYTOP:	Y-coordinate of the top line. Each line is 25 units vertical.

<u>CALL TRACK</u>

To enable the tracking cross and its control. Any light pen sensitive item may be picked with the light pen, attached to the tracking cross with the use of a menu button, and moved anywhere on the screen. TRACK is a Macro (assembler)-coded routine contained in module GT40TC. Before TRACK is called, the images to be manipulated must be displayed as light-pen sensitive items. When called, TRACK displays the following menu of options as light-pen sensitive items (light buttons):

ATTACH
DETACH
NO - X
NO - Y
X-AND-Y
RESTORE
END.

A tracking cross is also displayed.

To move an item, the item is first touched with the light pen, followed by hitting the ATTACH light button. When the tracking cross is touched, the first point of the item is attached to the center of the cross, with all other points of the item remaining in their original relationship to the first point. As the light pen is moved about the screen, the tracking cross follows the pen, carrying with it the attached item. When the item is located as desired, lifting the pen from the tracking cross and touching the DETACH light button causes the item to remain where it is located. The other items may be moved by repeating the above procedure.

Light button NO-X will cause the item attached to only follow the Y movement of the tracking cross. This is generally used when studying the first point of contact between a preform and its finishing die. NO-X allows only up and down motion, similar to what occurs in a forging operation. NO-Y limits movement to the horizontal axis only. If both NO-X and NO-Y are touched, no motion will occur. X-AND-Y restores full movement to the image.

RESTORE returns all images to the position they were originally created at. It is not necessary to use RESTORE before END. Figure VII-23 was made by using TRACK with NO-X to separate the die profiles from the preliminary preform image created by PFPREP. The preliminary image is shown in Figure VII-17a. After positioning the die profiles and returning to PREFRM without using RESTORE, the Auto function of PREFRM was used to modify the preform to the shape shown in Figure VII-17b. The plot function of PREFRM was then used to copy the die and preform images, and to create the unit vector.

The plot of the original finished shape was added to Figure VII-17 as follows. PREFRM was terminated, returning control to the root segment. The option Preform was then selected which called PFPREP. This time, no modifications were made to the preform and PREFRM was called directly by selecting the Done option in PFPREP. When PREFRM was entered, the Plot option was selected with no modifications being made to the preform. When PLOT was entered, only a copy of the preform was requested. Since, in the previous use of the Move option in TRACK, only the die surfaces had been moved, the parting line points of the preform and the finished part were aligned when plotted.

Subroutine INITDS

This subroutine initializes the graphics drivers and calculates the limits of the image to be drawn so the X and Z values of the image are in a 1:1 relationship with the X and Z values of the part.

The display window is the portion of the CRT where the cross section of the part will be drawn. The window is defined such that an area at the bottom on the CRT is reserved for text messages. The display window is defined in terms of raster units on the CRT and is referred to as object space. The aspect ratio is the ratio of the length of X to the length of Y of the defined window.

The coordinate system of the actual part is referred to as subject space. The center and half-ranges of the part are calculated in subject space. It is necessary to define the limits of the subject space so that when drawn in object space, the figure will appear as large as possible within the available window without distortion. This is done by adjusting the size of the subject space window so that the aspect ratio in subject space is the same as the aspect ratio in object space.

After calculating the center and half-ranges of the part window, the aspect ratio of the part window is compared to the aspect ratio of object space (the CRT window). If the part aspect ratio is less than the object space aspect ratio, the range in Z of the part is expanded to make the two equal. If the part aspect ratio is greater than the object space window, the range in X is expanded. Thus, the range of one axis of the part will extend the full range of this same axis in object space, with the range of the other axis being artificially expanded so as to retain a 1:1 relationship between X and Z of the part and its object space image.

Data is passed to and from this subroutine via the Labeled Common block ARRIGS, and the Blank Common block.

Subroutine INTRPL (XPLOT,ZPLOT,NPLOT,XPS,ZPS,RPS,IPS)

Given a surface defined by an array of fillet or corner coordinates and the radius associated with each, INTRPL will calculate the array of points needed to describe the radii by short-line segments. The first array of points (polygonal shape) is referred to as the part file. The second array of points (smoothed shape) is referred to as the display file. It is this latter shape which is displayed on the CRT, after being scaled and translated as necessary.

The fitting of line segments to a radius is done via subroutine FITARC. The minimum line segment is found by dividing the overall width of the section by 250. This results in a segment size equivalent to 4 raster units on the CRT. Because the display file has a fixed length, if an attempt is made to calculate more points than there is storage space available, the segment length is increased.

The routine starts by initializing the arc length to zero, the count of points in the display file to 1, and setting the first point in the display file equal to the first point in the part file. The minimum segment size is added to the arc length to establish its initial value.

A DO-loop is then started which generates the additional points needed to define the radii. A test is made of the absolute size of each radius. If it is less than the current arc length, it is considered too small to bother interpolating. Instead, the point in the part file defining the location of the radius is equated to the point in the display file. After each radius is interpolated, the number of points in the display file is compared to the maximum number allowable. If they are equal, the arc length is increased and the entire procedure restarted.

If the radius is large enough to require fitting of line segments, the interpolator subroutine FITARC is called. This operates by passing to it three points defining two lines, the radius at the intersection of the two lines, the name of the array where the interpolated points will be stored, the arc length to be used, and the maximum allowable number of points in the display file. The routine returns the interpolated points (values and count) and a flag indicating if an attempt was made to calculate more points than would fit in the output array. This flag is tested and if it is set, a branch is made to the beginning of the program where the arc length is increased. FITARC is called until all sets of three adjacent points have been interpolated.

The variables in the calling sequence are:

(1) XPLOT and ZPLOT: Arrays of the coordinates of the end points of line segments needed to display a curved figure as a series of straight-lines (output).

(2) NPLOT: The number of elements in the above arrays; maximum value = 500 (output).

(3) XPS and ZPS: Arrays of coordinates defining a polygonal shape (input).

(4) RPS: The array defining the radii at each vertex of the polygon (input).

(5) IPS: The number of elements in the arrays XPS, ZPS, and RPS (input).

Subroutine INT2LN (E1, E2, X, Y)

Given the coefficients for two lines, this subroutine finds the intersections of the two lines. The coefficients are passed via the three element arrays E1 and E2. The point of intersection is returned as X and Y.

Subroutine LEADER(N)

LEADER uses NCOUT to output leader code to the output file. N frames of leader code are generated. LEADER is called at the start of PRTSRF, before and after the title is punched, at the completion of each Y-Z machining pass, and at the end of PRTSRF just before the NC file is closed. The ASCII code for leader is equivalent to a line feed (12_8).

Subroutine LINEAR (X,Y,N,XI,YI,NI)

LINEAR is a subroutine that performs linear interpolation on a table of values X, Y using subroutine AITKN. The arrays X, Y of length N together with array XI of length NI are input. YI values are calculated, corresponding to each XI.

Subroutine LINEQ (X1, Y1, X2, Y2, COEF)

Given the coordinates of two points, X1, Y1 and X2, Y2, LINEQ determines the coefficients of the straight-line defined by the points. The coefficients are those for the general form of the equation of a straight-line which is A*X + B*Y = C. A, B, C are returned as the elements of the three element array, COEF.

Subroutine MARKIT (X,Z)

MARKIT draws a cross (+) on the CRT at the user coordinates X, Z. It is used to mark points of interest such as those indicated by a designer as the location of cavities.

When called, it first converts the X, Z coordinates from user space dimensions to the raster coordinates of object space on the CRT. It then calculates the ends of the cross, 10 raster units long on each side and centered about X, Z. Finally, it draws, as visible vectors, the horizontal and vertical lines of the cross.

Variables in the calling sequence:

X, Y: The coordinates of a point in user space where the cross is to be drawn (input).

Subroutine MAXMIN (A,AMAX,AMIN,IMAX,IMIN,N)

This subroutine determines the maximum and minimum values of an array, and their respective indices. It first initializes the maximum and minimum values to the first element of the array, and the indices to 1. It then tests all other values (from 2 to N) to see if the test value is a new minimum. If it is, it saves the current value being tested as the minimum, and the index of the value being tested. It then performs a similar operation to test for a new maximum.

Variables in calling sequence:

(1) A: A one-dimensional array of real values (input).
(2) AMAX: Maximum value found in the array (output).
(3) AMIN: Minimum value found in the array (output).
(4) IMAX: Index to the maximum value in the array (output).
(5) IMIN: Index to the minimum value in the array (output).
(6) N: Number of elements in the array to be tested (input).

Subroutine MERGE (XT,YT,NT,XS,YS,XI,YI,NS,XO,YO,NO,TOP)

This subroutine merges the points defining a part surface with the points defining the shear surfaces on this part. The result is a set of points defining a flow surface, along which metal will move when forging occurs with either friction or shear stress. Where a shear surface occurs, the four points defining the shear surface replaces all of the part surface points which lie above or below the shear surface (see Figure VII-11a, b, and c).

When called, MERGE first initializes indices for the die surface arrays, the shear boundary arrays, and the resultant flow surface arrays. The first point on the die surface is saved since it is always included in the flow surface. The upper bound for the index of the points on the die surface to test for equivalence with a shear boundary is set to one less than the total number of points defining the part surface. This is because the last point is always the extreme rightmost flash boundary and a shear cavity cannot lie under the flash.

A test is then made to determine if there are any shear boundaries along the surface. If not, all of the die surface points become the flow surface points in a one-to-one relationship. If shear surfaces do exist, the index to the die surface is incremented by one and a test performed to see if all die surface points have been tested. This test will never be true when first evaluated. When there are more die surface points to consider, a comparison is made between the current die surface point and the current shear surface boundary. This is done by comparing the X values of the two points. If the current die surface point has an X value less than the X value of the current shear boundary, the die surface point is added to the flow surface array, and the next die surface point tested.

If the current die surface point has an X value equal to or larger than the X value of the current shear boundary, the die surface point is discarded. The left shear boundary, the two shear surface points, and the right shear boundary are then added to the flow surface arrays. A loop is then started to eliminate all the die surface points which lie above (for the upper surface), or below (for the lower surface) the shear surface. This is done by testing and discarding die surface points which have X values less than the X value of the right-hand shear boundary of the cavity just found. When a die surface point is found with an X value greater than that of the most recent right-shear boundary, the index to the merged surface is increased by four to account for the four shear surface points added to the merged surface arrays. The index to the shear boundary points is incremented, and a test made to determine if all shear surfaces have been found. If they have, a branch is made to add the remaining die surface points to the merged surface arrays. If more shear surfaces exist, the current die surface point is added to the flow surface file, and the entire procedure is repeated.

When the generation of the merged flow surface is completed, subroutine FILTER is called to eliminate points which lie closer than 0.010 inches in X, and to reduce the number of points defining the flow surface to 50 or less. When this is completed, control is returned to FLWSRF.

Calling sequence arguments:

(1) XT,YT: Arrays defining the coordinates of the die surface (input).

(2) NT: The number of points defining the die surface (input).

(3) XS,YS: Arrays defining the coordinates of the shear surface boundaries (input).

(4) XI,YI: Arrays defining the coordinates of the points calculated to be on the shear surfaces (input).

(5) NS: The number of shear surface boundaries, plus one (input).

(6) XO,YO: Arrays defining the coordinates of the merged and filtered flow surface (output).

(7) NO: The number of points defining the flow surface (output).

(8) TOP: A logical variable indicating which surface (upper or lower) is being operated on (input).

Subroutine MOVEND

MOVEND is used to move the parting line points and the two points adjacent to each parting line by an amount specified. If the amount specified is positive, the parting line points will be moved toward each other. This is the usual case as it is normally desired to make the preform slightly narrower than the finish die. This ensures the preform will fit into the finisher.

After the designer enters the distance each end is to be moved, the program branches if the section is axisymmetric with single flash. If it is this type, moving the axis is not desired. If the section has flash on both sides, the first and last two points of the section have the amount entered added to their X-coordinates. It should be noted the last point is a repeat of the first point and is used so the display image of the preform will appear as a closed shape. The amount entered is then also subtracted from the X-coordinate of the right-hand parting line point and the first point adjacent to it on the top and bottom surfaces.

Data is passed to this subroutine via Labeled Common blocks DEFEL, SECTN, and SYSPR.

Subroutine NCOUT(IC,ID)

The subroutine stores all characters which are to be output to the NC tape file in a buffer. When the buffer is full, or when otherwise directed, the buffer is copied to a disk file.

When called for the very first time, NCOUT creates a direct access disk file on which the tape image will be stored. The user is asked to enter the name for this file. The character being output (IC) is then stored in the buffer using subroutine STORE. When the buffer is full (512 characters), the buffer is copied to the output file and then cleared. The buffer will be copied before it is full if parameter ID is set not equal to one.

The parameters in the calling sequence are as follows:

IC: The character to be output

ID: A flag. If set not equal to one, the buffer will be copied to disk after character IC is added to the buffer. In addition, the disk file will be closed after copying the last buffer.

Subroutine NEWRIB(L,M)

NEWRIB adds to the preform polygon the four coordinates needed to define a rib. The rib is added between the adjacent existing points L and M indicated by the user with the light pen. When called, NEWRIB first opens the existing array to allow the four new points to be added. The horizontal distance between the two indicated points is then found. The distance from each new fillet point to the indicated point adjacent to it is found as .1228 (i.e., tan 7) of the distance between the indicated points. The Z-coordinate of each fillet is found by using function XYINTR. The X-coordinates of the corners are set the same distance from their adjacent fillet as the fillets are from the indicated points. The Z-coordinates of the corners are set equal to the distance between the indicated points. The radii of the four coordinates are equal to half the distance between the corners, with signs as appropriate. The construction of the new rib is shown in Figure VII-12.

Calling sequence arguments: L and M are the indices to the first and second (in clockwise fashion) preform polygon points which bound the location of the new rib.

Data is also passed via Labeled Common blocks DEFEL, SECTN, and SYSPR.

Subroutine PARAMS

PARAMS determines the part and flash volumes and the plan area. It then prints, on both the CRT and the printer file, these values and the section properties calculated by XSECA.

If the part is in plane strain, the part volume is equal to the cross-sectional area times the depth. Referring to Figure VII-13, the flash volume is calculated as FLSVOL = 2* (flash width) * (flash thickness) * (depth) * (flash factor).

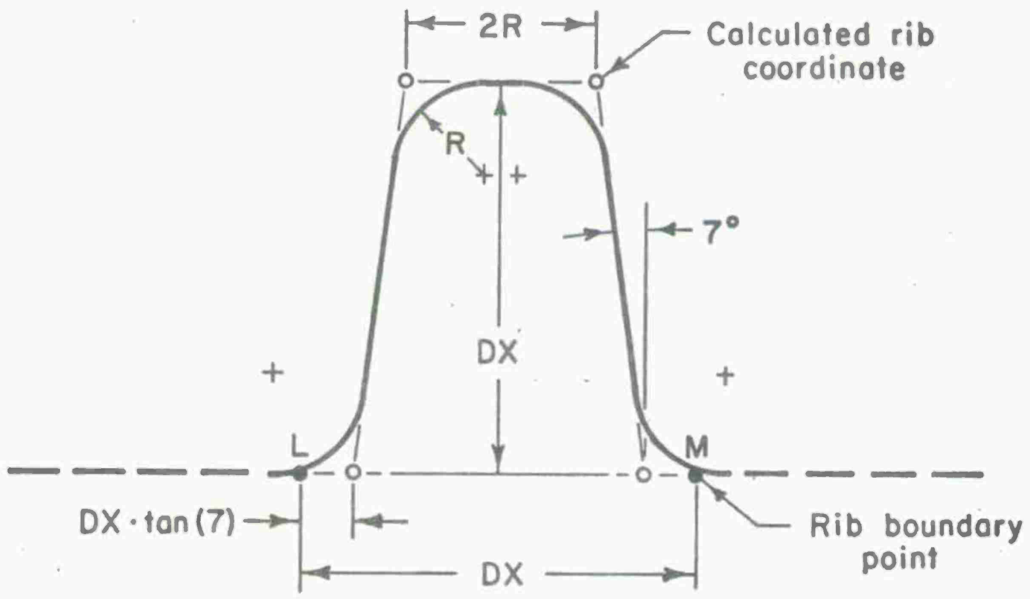


FIGURE VII-12. SUBROUTINE NEWRIB CREATING A RIB BETWEEN TWO EXISTING POINTS

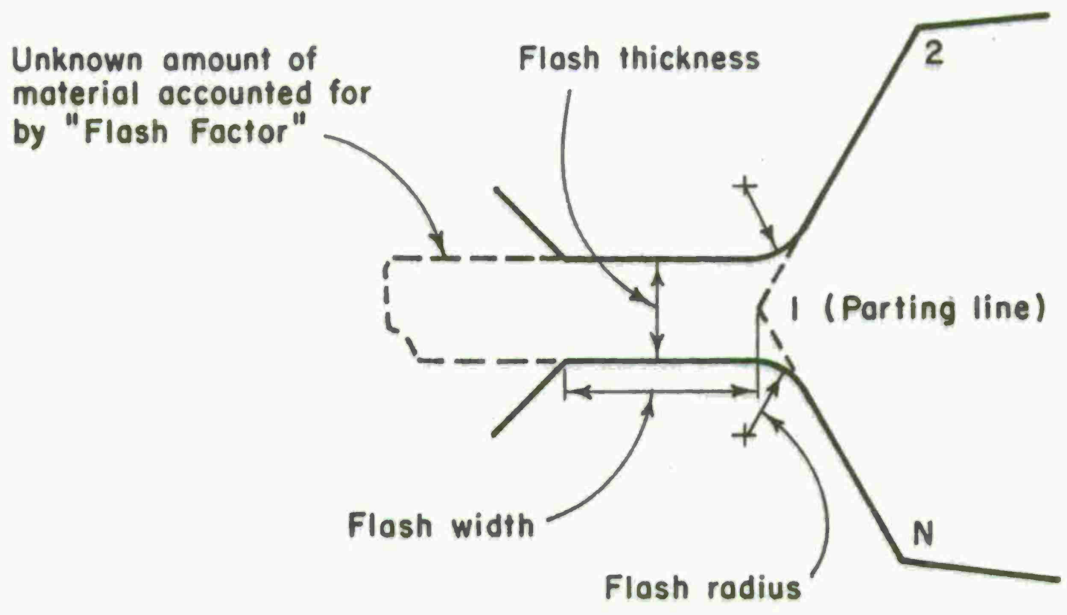


FIGURE VII-13. DEFINITION OF FLASH PARAMETERS

The flash factor is a judgment value used to account for the extra flash material squeezed out beyond the defined limits of the flash width. The flash volume is the volume defined as a rectangular solid by the flash dimensions multiplied by the flash factor.

It should be noted that a simplifying assumption is made with regard to calculating flash and volume (for either plane strain or axisymmetric cases); this assumption is that the inside coordinates of the flash are equal to the original parting line coordinate of the section, the Y coordinates are equal to half the thickness above and below the parting line Y value (see sketch), and the flash radius effects can be ignored. It is felt over the normal range of flash width and thickness values, these assumptions will not introduce any significant error. This is especially true when one considers that the flash factor value is a best-judgment estimate.

The plan area for a plane-strain section is the distance between the two parting-line coordinates, plus twice the flash width (flash in on both sides), all times the depth of the section.

Similar calculations are performed for axisymmetric sections, except the volumes and plan area are calculated based on areas of revolution about the axis of symmetry. The wedge angle entered in the section header card is used as the angle of rotation.

Data is passed to and from this routine via the Labeled Common blocks SYSPR and SECTN.

Subroutine PARTNO(NCHAR,NPART)

PARTNO formats an ASCII text string for the output file. This text string could be the part name or description, or instructions to the operator. PARTNO first generates an "M80" code which indicates to the NC controller that what follows is text and not vector data. The characters in the text string are then output one at a time by using a Do-loop from 1 to NCHAR. An end-of-block code is output following the text.

The parameters in the calling sequence are as follows:

NCHAR:	The number of characters in the text string which are to be output
NPART:	A text string (byte array) which is to be output.

Subroutine PFPREP

PFPREP allows the operator to add or delete points to the finish die polygon in order to prepare the polygon for the preform design operation. PFPREP is the first subroutine called when the "PREFORM" operation is requested.

When first entered, the routine copies the X, Z, and R coordinates of the finish die to new arrays created for the preform. In addition, a new, temporary last point is generated. This new point lies half way between the previous last point and the left-hand parting line point. The latter element is the first item in the arrays. By doing this, it becomes possible to indicate an intersection of two lines, where one of the lines is defined by the original last point and the left parting line point. Since points are selected with the light pen (LP), if the first point was simply copied as the last point, it would be impossible to distinguish between the two with the light pen as they would appear superimposed on each other.

Following this, the points are scaled and plotted on the CRT, LP sensitive. The upper and lower part surfaces are reduced in intensity and changed from solid to dashed-line construction. A list of options is then typed, each option being followed by a number in parenthesis. The system waits until the designer types the value corresponding to the function he wishes to have performed. The options are as follows:

(1) INTERSECT
(2) DELETE
(3) ADD POINT
(4) ADD RIB
(5) DONE.

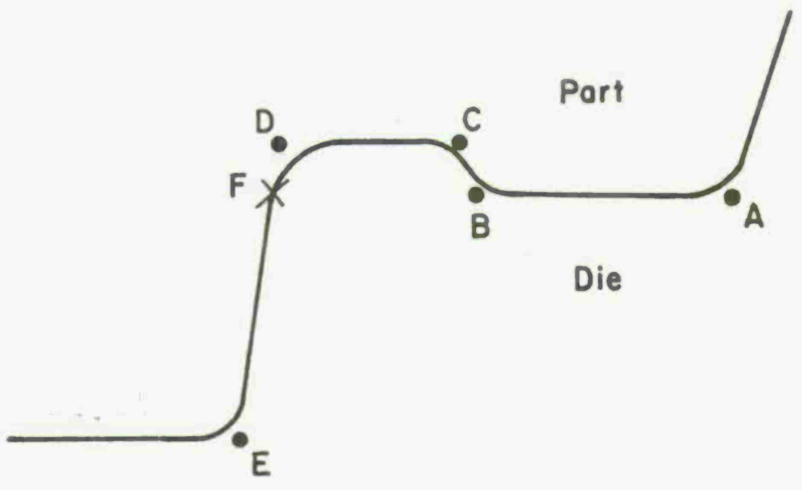
If option 1, INTERSECT, is requested, subroutine GETHIT is called with a parameter of four, indicating LP hits on four points are desired. This subroutine returns the array indices for the four points indicated by the designer.

The equations for the two lines defined by the two pairs of points are found using LINEQ, and the point of intersection of the two lines is determined using INT2LN. The operator is then asked to enter the value for the radius at the point of intersection. When doing this, he must also enter the correct sign for the radius: positive if a fillet, and negative if a corner. The X, Z, and R values for the new point are then inserted into their proper sequential position in the preform arrays. The program then loops, replots all the points, and again asks the operator to indicate which function is desired next.

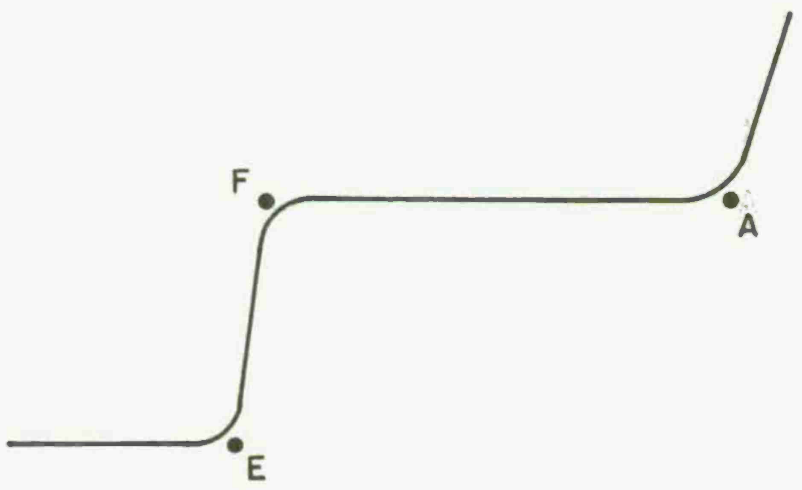
If "2" is entered by the operator, the program enters the "delete points" mode. The operator is told to indicate the points to be deleted with the light pen. When he gets a hit on a point, the point is marked with a cross (+). If it is the point he wishes to delete, he next hits the "ACCEPT" text with the L.P. When this is done, the point currently marked is blanked from the screen. As this is done, the value for the radius for the point is equated to a very large (10^{30}) value. He may continue in this manner until all desired points have been deleted. When this occurs, he then hits the "END" text with the L.P. which stops any further points from being accepted.

Following this, the preform arrays are compressed by GETRID which eliminates all points which have radius values equal to 10^{30}. The program then loops, replots all remaining points, and asks the operator to indicate what operation is to be performed next.

The primary intent of the INTERSECT and DELETE functions is to allow the operator to remove minor indentations from the profile of the preform. Such a situation is shown in Figure VII-14A. If the indentation described by points BCDE was to be eliminated, INTERSECT would first be used to generate the new point F. This is at the intersection of the lines defined by point pairs A, B, and D, E. A positive value radius would be specified indicating the new radius was to be a fillet. The function DELETE would then be used to eliminate Points B, C, and D. The result would surface AFE, shown in Figure VII-14B. Another "before" and "after" illustration of the use of these two functions is given in Figure VII-15.



A. Before processing with "PFPREP"



B. After processing with "PFPREP"

FIGURE VII-14. FUNCTIONS "INTERSECT" AND "DELETE"

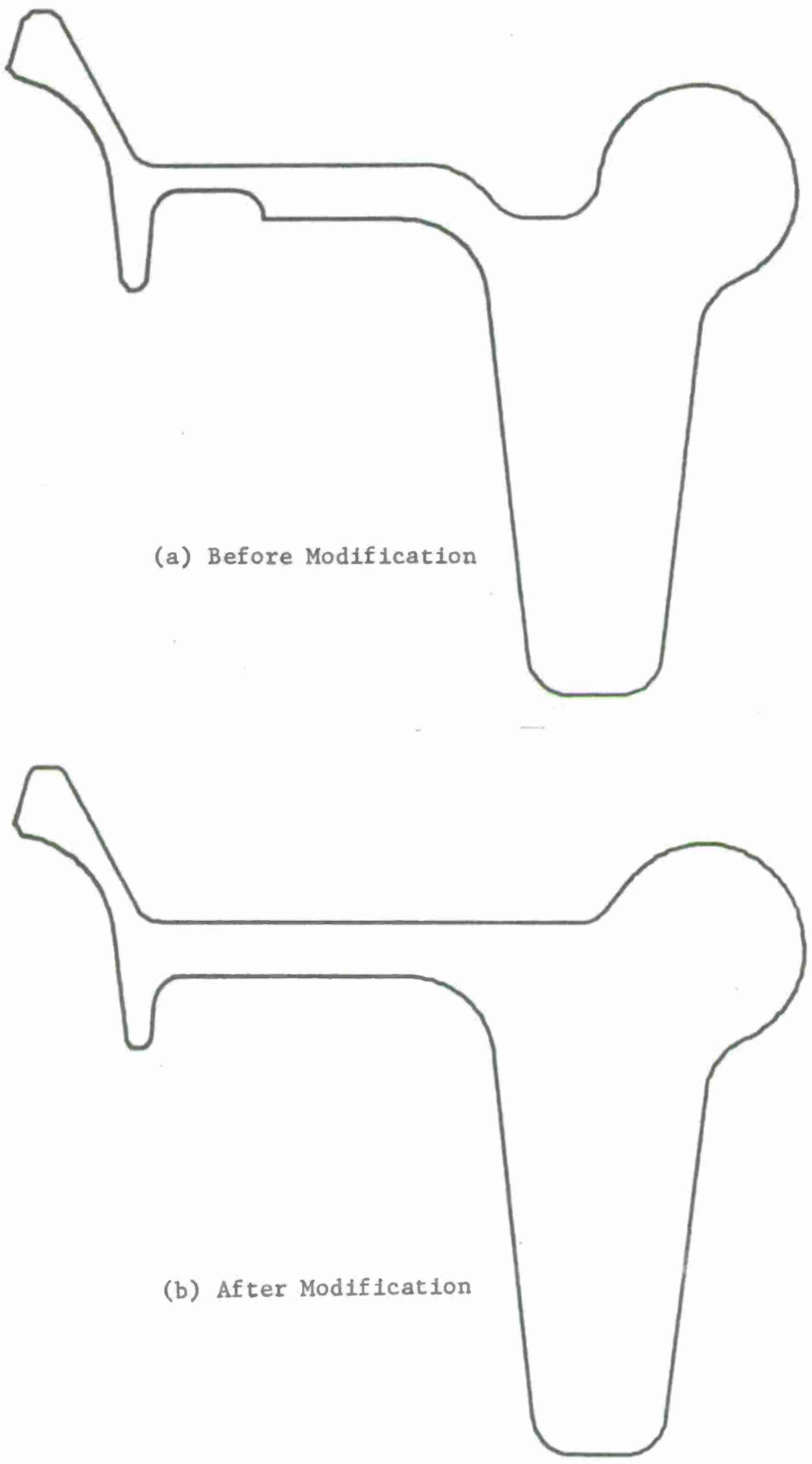


FIGURE VII-15. USE OF INTERSECT AND DELETE FUNCTIONS TO MODIFY A PREFORM POLYGON (SUBROUTINE PFPREP)

A new point can also be added anywhere along the surface of the preform by entering a "3" in response to the program's request for guidance. When this is done, the coordinate points are erased, and the two surfaces representing the finish die are made L.P. sensitive at a high brightness level. The operator is then instructed to indicate the location of the new point with the L.P. The point indicated is marked with a "+", with the marker following the L.P. along the surface until "ACCEPT" is hit. When this is done, the coordinates of the point indicated at the time are returned. The surface being indicated (upper or lower) is determined by the item tag value returned by the subroutine GETHIT. Knowing which surface the point was on and the X-coordinate of the point, subroutine ADDPT is used to create and insert the new point into the preform array.

One of the purposes of this function is to allow a web surface to be changed from a straight-line surface to a surface with a bevel. This would permit the finishing die to contact the preform at a relatively small spot, and start the metal flowing, in a gradual manner, rather than having the die contact a large area simultaneously. The before and after effect of this function is given in Figures VII-16 and VII-17. This function can also be used to create temporary points. Temporary points are used to create other elements but are not, themselves, part of the permanent preform polygon.

A new rib can be added typing a "4". One use of this feature is to allow a rib to be added opposite an existing rib to prevent a "suck-in" defect. If this extra rib is not added, the extrusion effect when finish forging the existing rib may cause the material to suck-in, thus leaving a cavity or dimple effect in the web behind the rib. A new rib can also be used to add additional material to a preform wherever it is needed. The operator is asked to indicate if the new rib is to lie opposite an existing rib. If he responds by typing "Y" (yes), subroutine ADDRIB is used to create the new rib, based on the size of the rib indicated by the designer with the light pen. If the designer responded to the inquiry with anything other than "Y", the new rib will be added to a web between two existing points. GETHIT is used to find the two existing points. If they are found to be adjacent to one another, subroutine NEWRIB is then used to construct the rib.

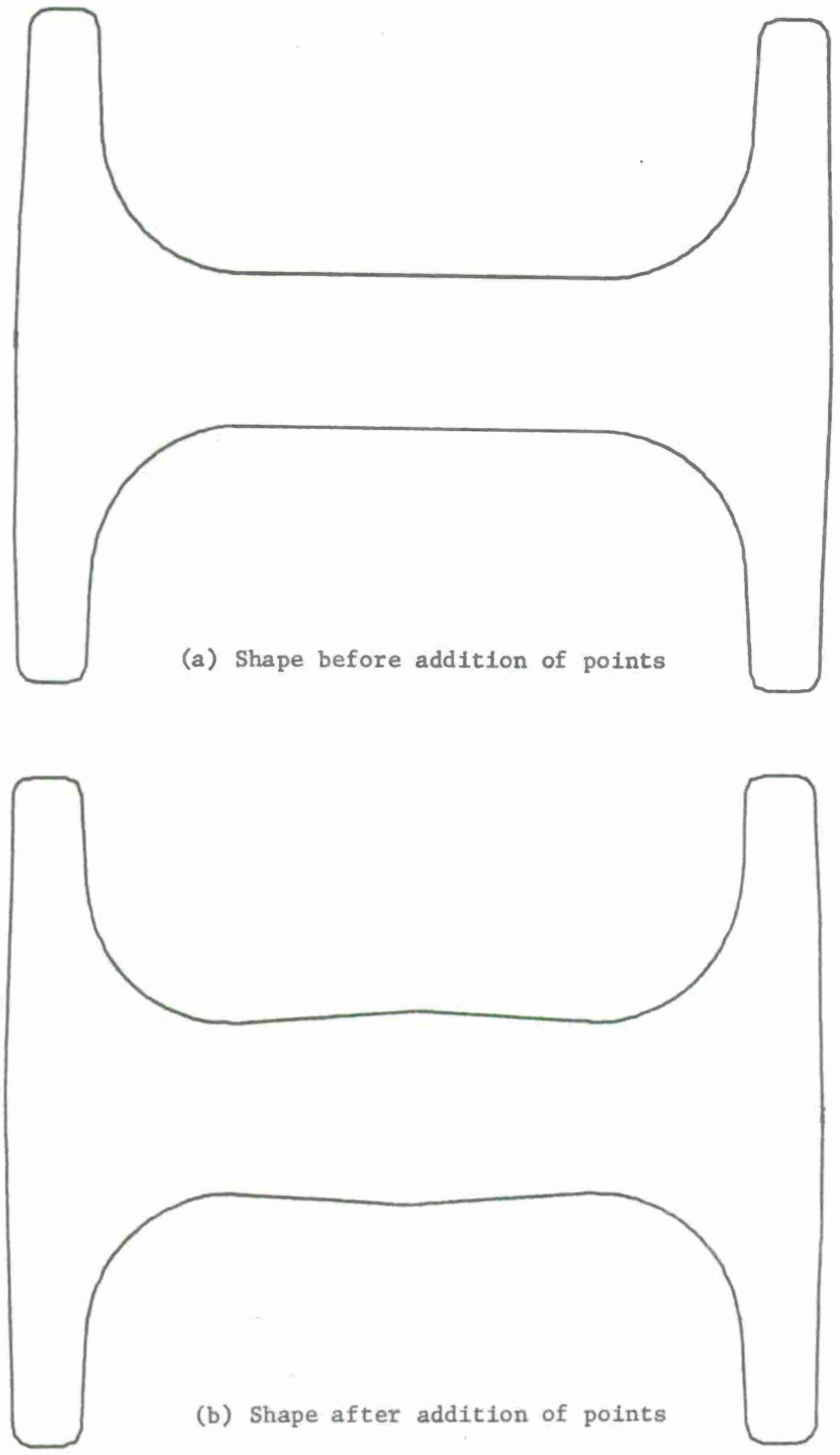
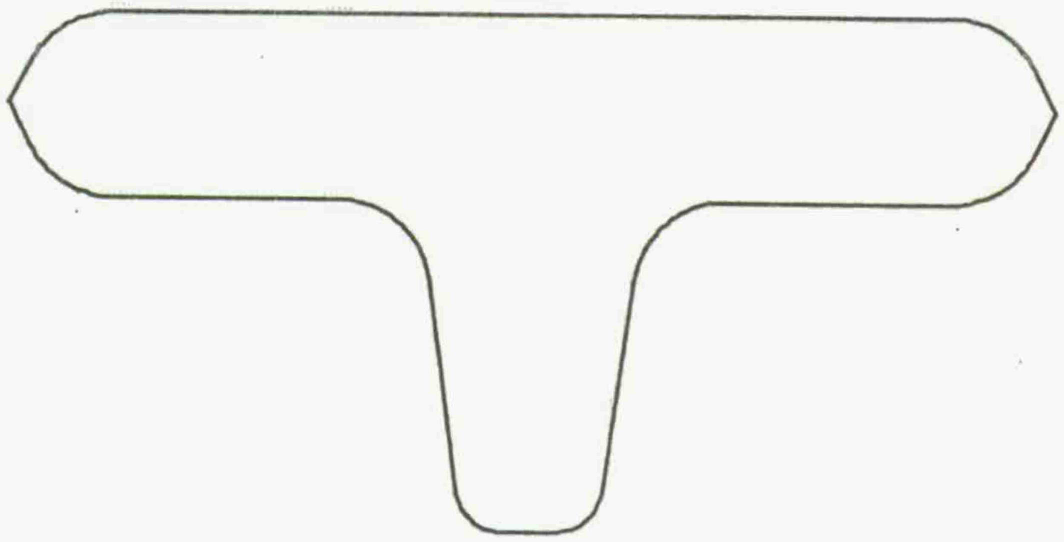
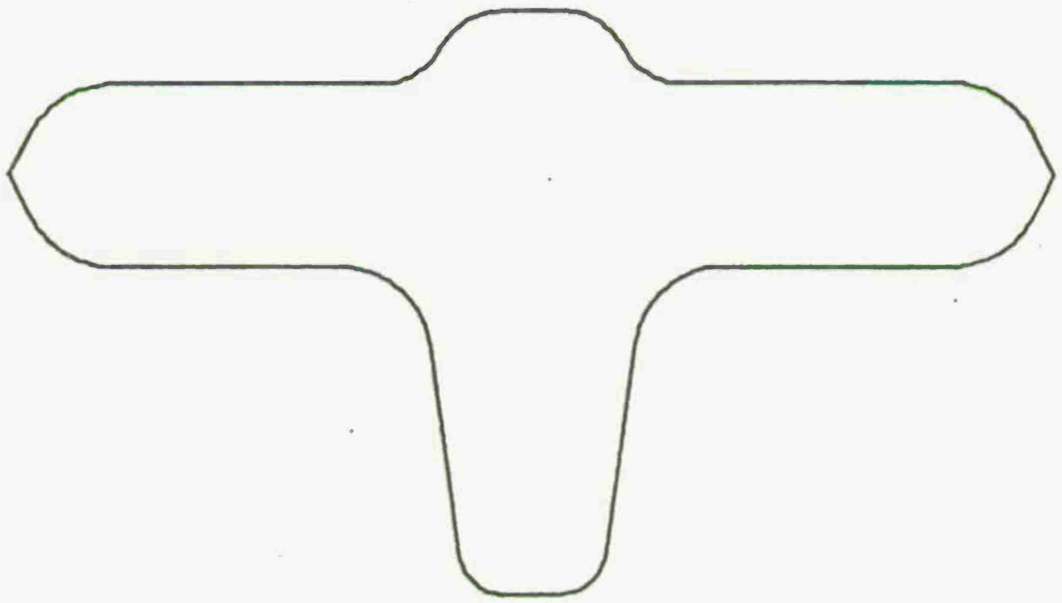

(a) Shape before addition of points
(b) Shape after addition of points

FIGURE VII-16. ADDITION OF POINTS TO CHANGE A WEB FROM FLAT TO BEVELED, SUBROUTINE PFPREP



(a) Shape before addition of rib



(b) Shape after addition of rib

FIGURE VII-17. ADDITION OF A RIB TO PREVENT SUCK-IN DEFECTS, SUBROUTINE PFPREP

When all points desired have been added or removed, the designer indicates he is completed with this preprocessing operation by typing "5". The array of radii for each coordinate point in the preform are then searched for three consecutive corners (positive values). This is because subsequent preform design assumes ribs are defined by only two consecutive corners. If there are three adjacent corners found, the third corner is marked with a cross (+), and the program pauses. When the user continues the program, it loops back and asks what operation is to be performed next. The operator would be expected to delete one of the three corners, or make some other modification which would eliminate the condition of three consecutive corners.

When the condition of not more than two consecutive corners is satisfied, subroutine GETRID is used to eliminate temporary construction points, and then control returns to TRKFRG. Before exiting, the arrays defining the preform polygon are written to the printer file, if the printer control variable has been set.

Data for this subroutine is passed via Labeled Common blocks, SYSPR, SECTN, DISPLA, and DEFEL.

Subroutine PICTUR

PICTUR is used to generate the display file of the upper and lower dies and to display these images on the CRT. The images include the flash geometry on one or both sides as appropriate. PICTUR does no computation directly; instead, it calls subroutine DIESRF, INTRPL, and INITDS. After issuing these calls, the part outline is drawn. Data is passed to or from PICTUR via the Labeled Common blocks SECTN, ARRIGS, PSEUPO, and STRSPT, and Blank Common.

Subroutine PLNSLD (FS,SIGVE,HE,HB,WID,SIGYB,P)

PLNSLD is used in energy calculations to find the stress and load on an element in plane strain converging flow. The details of the equations used are given in Appendix V.

Subroutine PLOTER

PLOTER is used to create hard-copy plots on the Hewlett Packard X-Y recorder. It links the coordinate data for the item specified from the display file to the plotter drivers. These drivers then output the data to the recorder via the LPS analog outputs. When started, it first sets the plotter scale factor to 1.0, which assumes the plot will be to the same scale as the image on the CRT. The overall width of the real section is tested against a value of 10.0. 10.0 inches is the maximum travel of the plotter when using a gain of 1 volt/inch. If the real section is less than 10-inches wide, the operator is asked to indicate the type of scaling: F = full (actual) scale; S = screen scale. The default value is S. If S is indicated (by typed entry or default), the plotted image will be as large as possible on the 8-1/2 x 11 paper.

Once the scale is set, the operator is asked to indicate the type of plot desired. If he enters a "1", the profiles of the top and bottom finish dies are drawn. If he enters a "2", the complete preform profile is drawn. If he enters a "3", a unit vector is drawn and the routine terminates. The unit vector allows the scale of the plot to be determined, since the scale is not otherwise noted on the plot. Figure I-9 and I-10 are typical plots made by this subroutine.

Data is passed to this routine via the Labeled Common block SYSPR and DISPLA.

Module PLOTOB

This module consists of four FORTRAN-callable subroutines which allow the user to control the X-Y recorder. Plots can be made by copying items from the display file used by the GT-40 graphics CRT display, or by directly plotting from a data base. The plotter has a range of Ø to 4095 in both X and Y. With the recorder set to a gain of 1. Volt/inch, this gives a maximum plot size of 10 inches on both axes.

Summary of FORTRAN-callable subroutines in PLOTOB:

CALL COPYGT (ITAG,SCALE)

This call copies the item specified by ITAG from the graphics display file to the X-Y plotter. If SCALE equals 1.0, the plot will be to the same scale as the CRT image. If SCALE is less than 1.0, the plot will be smaller than the CRT image.

CALL ENDXY

To complete the plotting process and turn off the plotter interface. This should be the last call to the X-Y plotter. It should always be issued before writing any text on the CRT after having used COPYGT.

CALL INITXY (IX,IY)

Initializes the X-Y plotter and leaves pen at IX, IY (object space units). Gain adjustments can be made at this time. This must be the first call to the X-Y plotter.

CALL PLOTXY (IX,IY,IP)

Draws a vector from the present position of the pen to IX,IY. IP = Ø for pen up, IP = 1 for pen down.

Subroutine PLSTRS (I1,IZ,FS,FD,SIGYB,SIGYE,P,XCG,FLAG)

PLSTRS finds the ending stress, load, and center of load for a deformation element in plane strain. A complete explanation of the technique used to find these values is given in Appendix III.

The variables in the calling sequence are as follows:

(1)	I1	:	The index to the left boundary of the element (input).
(2)	I2	:	The index to the right boundary of the element (input).
(3)	FS	:	The flow stress of the material; taken to be the yield strength at the forging temperature and rate (input).
(4)	FD	:	+1 if material flow is left to right (i.e., scanning from right to left); -1 if flow is right to left (i.e., scanning from left to right) (input).
(5)	SIGYB	:	The ending stress on the element (output).
(6)	SIGYE	:	The beginning (known) stress on the element (input).
(7)	P	:	The vertical load on the element (output).
(8)	XCG	:	The center of loading on the element (output).
(9)	FLAG	:	A logical variable used to indicate that load calculations are to be made (input).

Subroutine PNCHNC(X,Y,Z,M)

PNCHNC formats absolute coordinate data into the incremental form required for the BCL CNC milling machine. To do this, it first converts the absolute value to an integer representing the size of the move in thousandths of an inch. That is, for an X-axis move

$$IX = \text{integer part } (1000 * X) \ . \qquad \text{(VII-21)}$$

This integer, absolute value, is then converted to integer incremental form by subtracting the current value from the previous value as follows:

$$IDX = IX_{new} - IX_{old} \ . \qquad \text{(VII-22)}$$

The old integer value is then replaced by the new integer value, and subroutine PXYZ is called to output the current move. When m = -1, the initial "old" values for the three axis are set, but no data is output.

After each set of X, Y, Z coordinates is processed, an end-of-block mark (i.e., carriage return, line feed) is appended to the output file using NCOUT.

To summarize the operation of the NC formatting routines, assume the following three(3) sets of absolute coordinates exist:

X	Y	Z
1.154	2.837	0.549
1.289	2.956	0.650
1.508	3.254	1.064

The two(2) vectors represented by these values would appear in the output file as:

X135	Y119	Z101	(first vector)
X219	Y298	Z414	(second vector)

It should be noted that PNCHNC contains data tables for preparing tape in either EIA or ASCII format. By "commenting" the one table not desired, either format can be produced.

The variables in the calling sequence are as follows:

X,Y,Z:	Arrays of absolute coordinates in user units
M:	The number of elements in each of the above arrays.

Subroutine PREFRM

Starting with the results from PFPREP, this subroutine allows the designer to direct how the shape is to be further modified to yield the preform section. One of the features is an "automatic" mode which will reduce and round all ribs, expand all non-rib radii, and adjust the webs as necessary to balance the volume. Although it does not guarantee a solution, this automatic function appears to work very well for plane-strain sections. It has had mixed success operating on axisymmetric sections.

When started, PREFRM first expands the total volume of the finish part, including flash by 3%. This new value will be used as the base when testing to see if the preform volume is satisfactory. The value used ensures that the preform volume is always larger than the total finished part volume. The first point in the coordinate arrays is then copied as the last point. This is followed by the identification of all ribs by using subroutine RIBID, and the determination of the index to the right-hand parting-line coordinate. The preform polygon, as generated by PFPREP, is then processed by DECUSP, VOLUME, INTRPL, and SCALZZ. This results in the profile of the preform being displayed on the CRT, superimposed on the finish die profiles.

A list of options is then typed, each option being followed by a number in parentheses. The system waits until the designer enters the number corresponding to the function he wishes to have performed. The options are as follows:

AUTO(1)
VALUES(2)
BALANCE(3)
RIBS(4)
WEBS(5)
RADII(6)

ENDS(7)

POINTS(8)

MOVE(9)

SAVE(10)

PLOT(11)

DONE(12).

If the designer enters any value or character other than those above, the program loops and retypes the options list.

If "1" is entered, the program enters the automatic design mode. It should be noted that a review of the literature yielded no consistent set of quantitative relationships regarding how a preform should be derived from a finished section. Qualitative descriptions, such as smoothing and blending, were what the literature primarily suggested. As a result, the relationships given below are not based on any body of analytical or empirical knowledge. They are based strictly on the author's limited experience and association with people involved with forging die design and forging operations, and his sense of esthetics.

The first operation performed in the automatic mode is the expansion of all non-rib radii. This is done by calling subroutine RADEXP. Once the non-rib radii are expanded, the next operation is to reduce the height of all ribs. For ribs which have been identified as such, and which have a rib ratio greater than 1.0, a new rib height is found from the following function:

$$RH' = .5\ RH\ (1 + e^{-RR})\ , \qquad \text{(VII-23)}$$

where RH is the original rib height, RR is the rib height-to-width ratio, and RH' is the modified rib height. RH'/RH as a function of RR is shown in Figure VII-19. The coordinates of the two new corners are found from the intersection of the new rib height with the original sides of the rib. The radii for the rib corners are set equal to one-half of the new top width. This results in a small flat spot on the top of the rib. This is felt to be an acceptable situation, as most ribs which this function is expected to encounter have relatively steep sides (6 - 8 degree draft angle). Figure VII-20 illustrates the above procedure for modifying a rib.

With the non-rib radii expanded (adding volume) and the ribs reduced (subtracting volume), the next step is to adjust the webs in such a manner that volume constancy is maintained between the preform and the finish die. This is done by using an iterative "bracket and halve" technique. An initial guess is

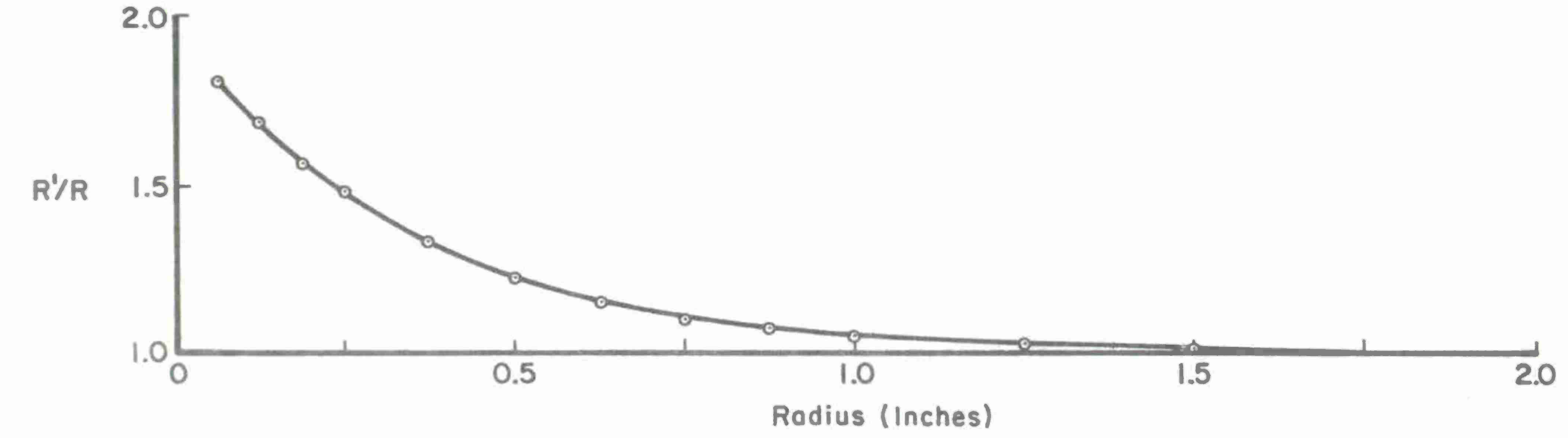


FIGURE VII-18 RADIUS EXPANSION RATIO

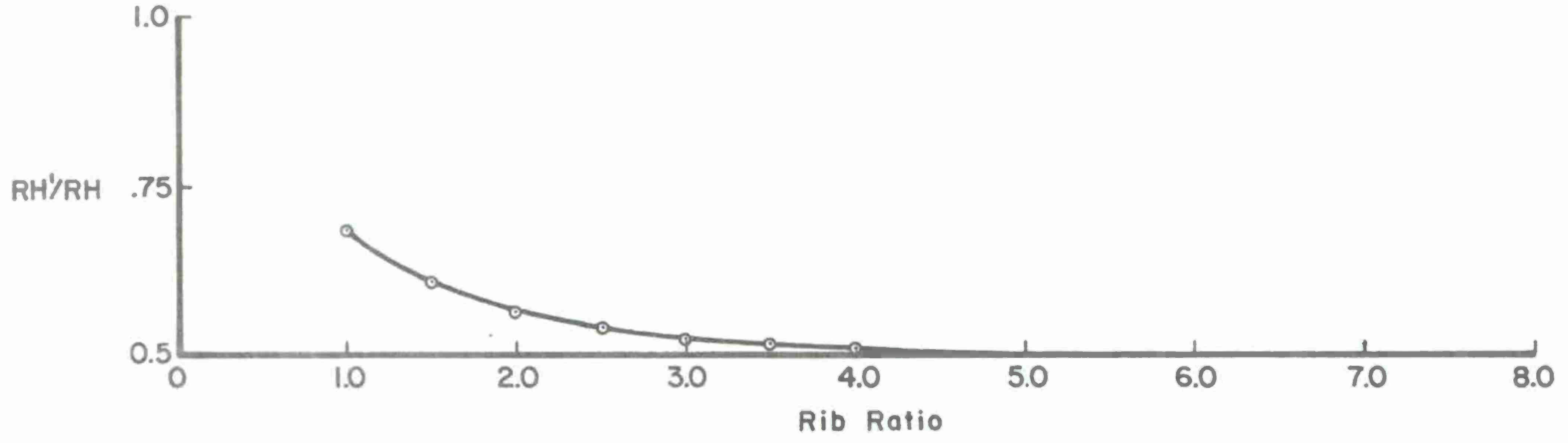


FIGURE VII-19. RIB REDUCTION RATIO

$$RH' = f\left(2\,RH/(RWB + RWT)\right)$$
$$R = RWT'/2$$

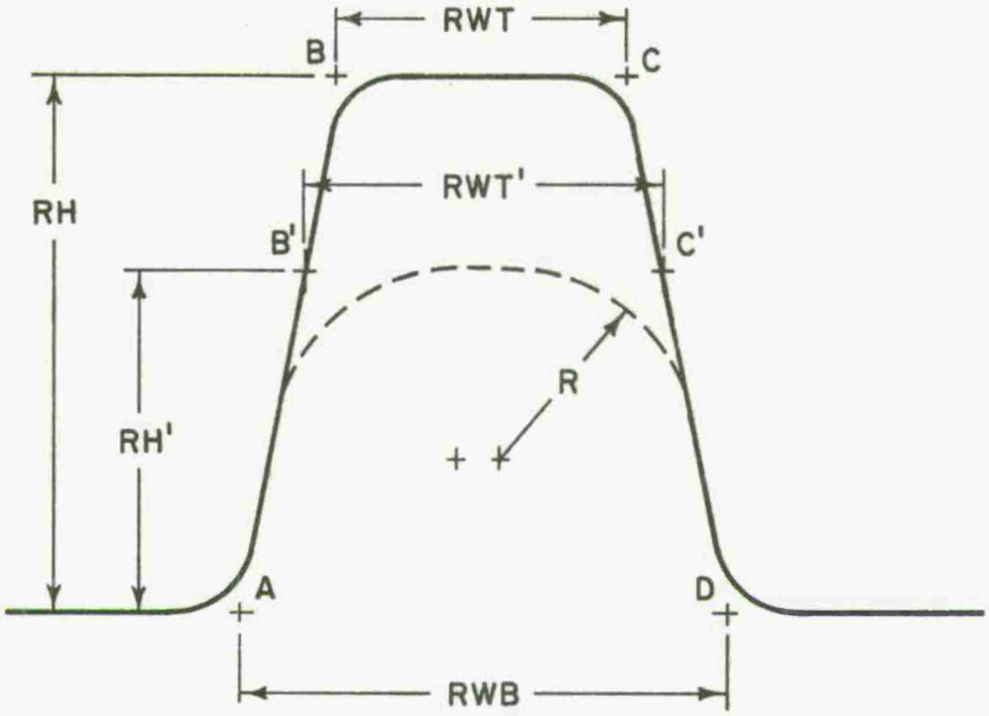


FIGURE VII-20. SUBROUTINE PREFRM - PROCEDURE FOR AUTOMATIC MODIFICATION OF RIBS

made as to the magnitude of the web change to make. The direction of change is determined by comparing the preform volume to the finish volume. If the preform is larger than the finish part, the webs need to be reduced. All web points are expanded or contracted by the amount of the initial guess. The new volume difference is calculated, and if greater than the allowable tolerance, the previous guess is halved, and sign set as appropriate, and another volume calculation made. This process continues until the difference is less than the tolerance, or until 10 trials have taken place. If the volume cannot be balanced after 10 trials, the balancing algorithm gives up and types a message that is unable to achieve a volume balance.

For the purposes of achieving a volume balance, all points, except the corners of identified ribs and the parting lines, are modified. For a web point which is not the fillet of a corner, the current value for the amount of change to make is added or subtracted, as appropriate, to the Z-coordinate of the point. If the point is a rib fillet, this same change is made to the Z-coordinate. In addition, however, the X-coordinate is also modified so that the modified point lies on the same line that forms the side of the rib that the original point lay on.

Once the balancing process is completed, whether successfully or not, the resulting polygon is processed by subroutine DECUSP to remove any cusps which may have been generated. The volume is then calculated, and the figured interpolated and displayed as a solid line image of relatively high brightness. The upper and lower finishing dies are also displayed, but as dashed lines at low intensity. The values for the preform volume, total finish volume, and the difference between these two values is typed, and the list of options is again displayed.

If Option 2, Values, is selected, the preform image is dimmed; the finish die images are blanked completely, and the individual coordinate points of the preform polygon displayed as light-pen sensitive dots. The designer is then asked to indicate two points. This is done with subroutine GETHIT. After he accepts the second point, the X, Z, and R values for all points between and including the two he indicated are typed.

If Option 3, Balance, is selected, the program attempts to balance the preform and finish volumes. This is done by branching to the same code used for volume balancing when in the automatic mode.

If Option 4, Ribs, is selected, the ribs previously found by subroutine RIBID are marked by a light-pen sensitive dot at the first corner of each rib. The operator is then asked to indicate the rib he wishes to modify. After identifying the rib with the light pen, he is asked to enter the rib height expansion ratio. The valid range of this ratio is between 0.3 and 2.0. The ratio would be less than 1.0 if the rib height is to be reduced. The height of the existing rib is determined by RIBPRM. The new rib height then becomes the product of the original rib height and the rib expansion ratio entered by the designer. Subroutine RIBFIL finds the coordinates of the modified rib corners, and RIBRAD is used to calculate the rib-corner radii. When this function is completed, the volume is calculated, and the new preform image and the original die surfaces are displayed as previously described.

If, when the rib expansion ratio was requested, nothing or zero was entered by the designer, the system would then request an amount by which the rib is to be thickened or thinned. If a positive value is entered, the flanks of the rib will be drawn in by the amount specified, thus thinning the rib. If a negative value is entered, the rib will be thickened by the amount specified. The absolute value of the amount specified cannot be greater than one-fifth of the width across the top (at the corners) of the existing rib.

The current preform volume is then saved. After adjusting the rib flanks as specified, the top of the rib will be raised or lowered as necessary to maintain volumetric balance before and after the change. Referring to Figure VII-21, the approximate change in volume is found as:

$$\Delta \text{ Volume} = DX\ (DY1_1 + DY2_2) \quad , \qquad \text{(VII-24)}$$

where DX = amount to thin rib

DY_1, DY_2 = heights of the rib sides.

The four X-coordinates of the rib points are then shifted horizontally by the rib thin amount (DX). The top width of the modified rib is found and used to estimate the amount the top must be raised to keep the volume in balance. This value is found as:

$$DZ \simeq 2* (\Delta \text{ Volume}/RWT') \quad , \qquad \text{(VII-25)}$$

where DZ = estimated change in vertical height of the rib

RWT' = width of the modified rib.

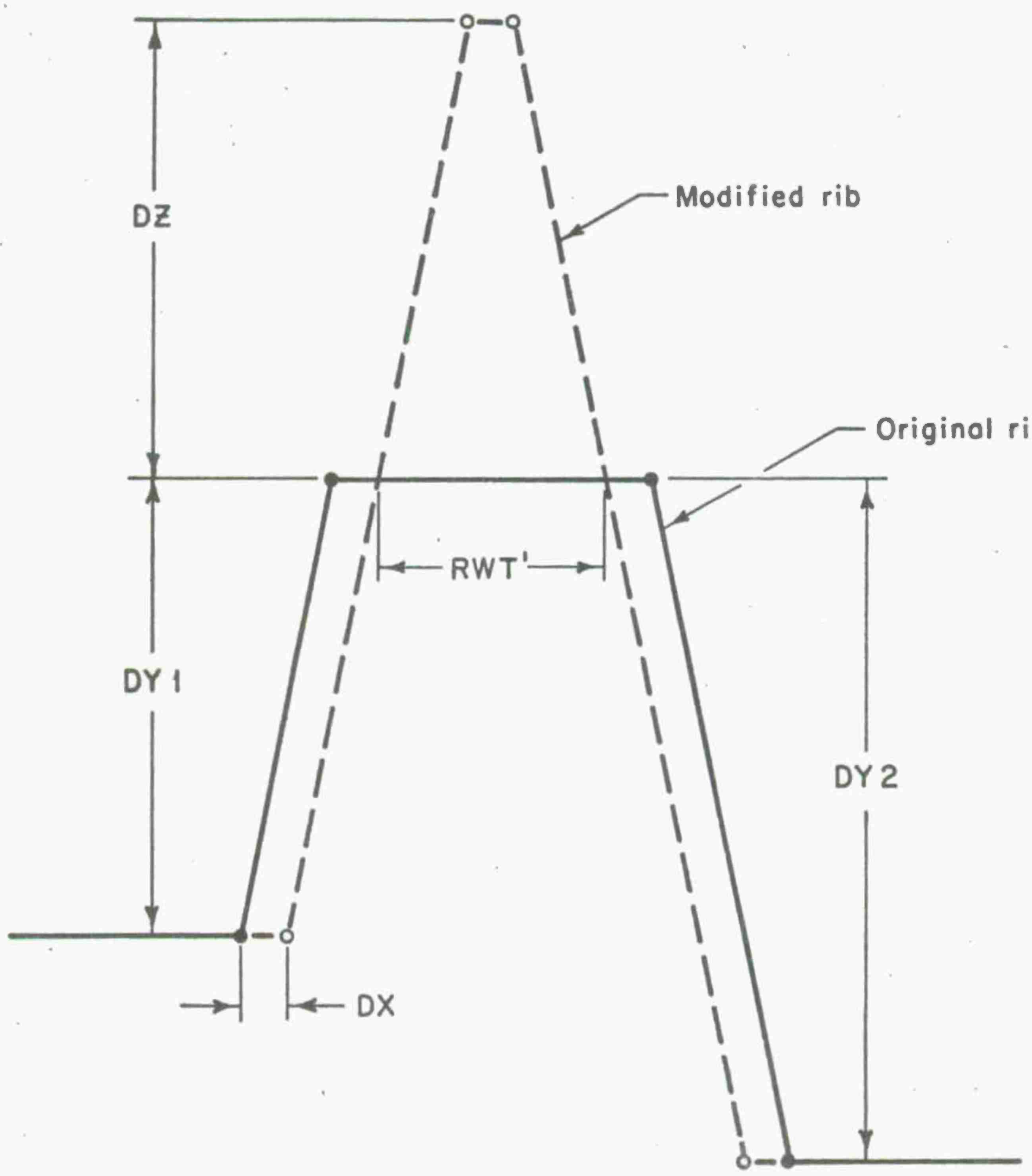


FIGURE VII-21. SUBROUTINE PREFRM. THINNING A RIB AND INCREASING ITS HEIGHT TO MAINTAIN VOLUME CONSTANCY

The Constant 2 is used to ensure the estimate is larger than actually needed. The estimated change in height is used by RIBFIL to find the modified corner coordinates, with RIBRAD used to determine the radii of the corners. VOLUME is then used to find the volume of the section with the modified rib. If the volume difference is less than one percent between the original section and the section with the modified rib, the routine terminates. If the difference is greater than one percent, the amount of the rib height change is halved. The sign of the next change in the rib height is set according to whether the modified section is larger or smaller than the original section. The process is then repeated. If the volume cannot be balanced after 10 trials, this fact is reported and the routine terminates.

When the rib thin amount was requested, if nothing or zero was entered, the above operations for thinning a rib while maintaining volumetric balance would be skipped. Instead, an absolute amount by which the top of the rib is to be raised or lowered is requested. If a positive value is entered, the height of the rib will be increased. A negative value will decrease the rib height. The absolute amount of the change cannot be greater than 25 percent of the existing rib height.

If Option 5, Webs, is selected, all of the coordinate points are displayed as light-pen sensitive dots, and the designer asked to indicate two web points. When the second point is accepted, the points returned are tested to see that they both lie on the same surface. If they are on the same surface, the points are then tested to determine if either point specified is the corner of a rib, or if any ribs lie between the two points. If either of these situations exists, the web modification process is terminated, and the program requests a new option. If the points indicated are valid web points, the designer is asked to enter the amount of the web change. If a positive value is entered, the volume will be increased; if a negative value is entered, the volume will decrease.

The two points indicated are then tested to determine if they are the trailing and/or leading fillet of a rib. If they are, the Z-coordinate of each is modified by adding or subtracting the web change amount, with the X-coordinate being found by subroutine RIBFIL. If the two indicated points are not rib fillets, only the Z-coordinate is adjusted. Regardless of the nature of the two indicated points, all other, if any, points between the two have the web change amount added or subtracted to the Z-coordinate of each.

If Option 6, Radii, is selected, the designer is asked if he wishes to modify all non-rib radii. If he responds by typing "Y" (yes), subroutine RADEXP is called and expands all non-rib radii by an inverse exponential function. If the designer gives any response other than "Y", the polygon coordinate points are displayed as light-pen sensitive dots. The operator is then asked to indicate the single radius to be modified by touching the light pen to the appropriate point. After he accepts a point, he is asked to enter the radius expansion ratio. The new radius then becomes the product of the original radius and the expansion ratio.

If Option 7, Ends, is selected, subroutine MOVEND is called. This allows the designer to move the parting line and adjacent points toward or away from each other by a specified amount.

If Option 8, Points, is selected, the polygon coordinate points are displayed as light-pen sensitive dots, and the designer asked to indicate the points to be modified. Any single point, or group of up to six adjacent points, may be moved with this function. If only one point is to be moved, it must be selected twice with the light pen. After the points are accepted, the operator is asked to enter the displacement change for X and Y. This is done by typing the two values on the same line, with a comma separator. The signs associated with each value controls the direction in which the point will be moved and follows a standard Cartesian coordinate system. That is, plus X moves the points to the right; plus Y moves the points up.

It should be noted that with any of the above routines, when the operator is asked to enter data values, the values may either be in whole number or decimal format. If an integer value is to be entered, it is not necessary to include the decimal point.

If Option 9, Move, is selected, the preform and finish die surface images are all displayed in solid-line form, moderate intensity, and light pen sensitive. Any text on the screen is eliminated by issuing five line feeds. The graphics tracking cross module, TRACK, is then called which allows the designer to move the three images relative to each other. The system has the ability to move an image only in the X-Y plane of the CRT screen. An image cannot be rotated.

One of the primary uses of Move is to allow the designer to visually check the way in which the preform will nest into the finishing dies and to determine where the first point of contact will occur. If he does not like the results, he may terminate the Move function and make further modifications to the preform using the functions above. Another use of Move is to arrange the images prior to making a hard-copy plot of them.

If Option 10, Save, is selected, the current definition of the preform polygon is saved as a disk file. This operation is handled by subroutine SAVPLY.

If Option 11, Copy, is selected, the preform and finish die images may be copied to the X-Y recorder. Subroutine PLOTER is called to handle the actual details of making the copy. Figure VII-22 and VII-23, among others, were made using this function. Figure IV-22 is the preform design for a section superimposed on the finish dies for this same section. The preform was created by first using the automatic function to do most of the work. The Radii function was then used to expand several of the radii an additional amount. The preform design for the "T" section in Figure IV-23 was made using only the automatic mode.

If Option 12, Done, is selected, the preform design module terminates and control is returned to the root segment, TRACK.

All data for PREFRM is passed to it using the Named Common blocks SYSPR, SECTN, DEFEL, and DISPLA.

Function PRLFLW (F,FR,SIGYE,T,W)

PRLFLW returns the value for the beginning stress on an element in plane strain between horizontal platen. The beginning stress, σ_{yb}, on an element is given by:

$$\sigma_{yb} = 2 * F * FR * W/T + \sigma_{ye} \quad , \qquad \text{(VII-26)}$$

where σ_{ye} = ending stress on element

F = friction factor

FR = material flow stress

W = width of section

T = thickness of section.

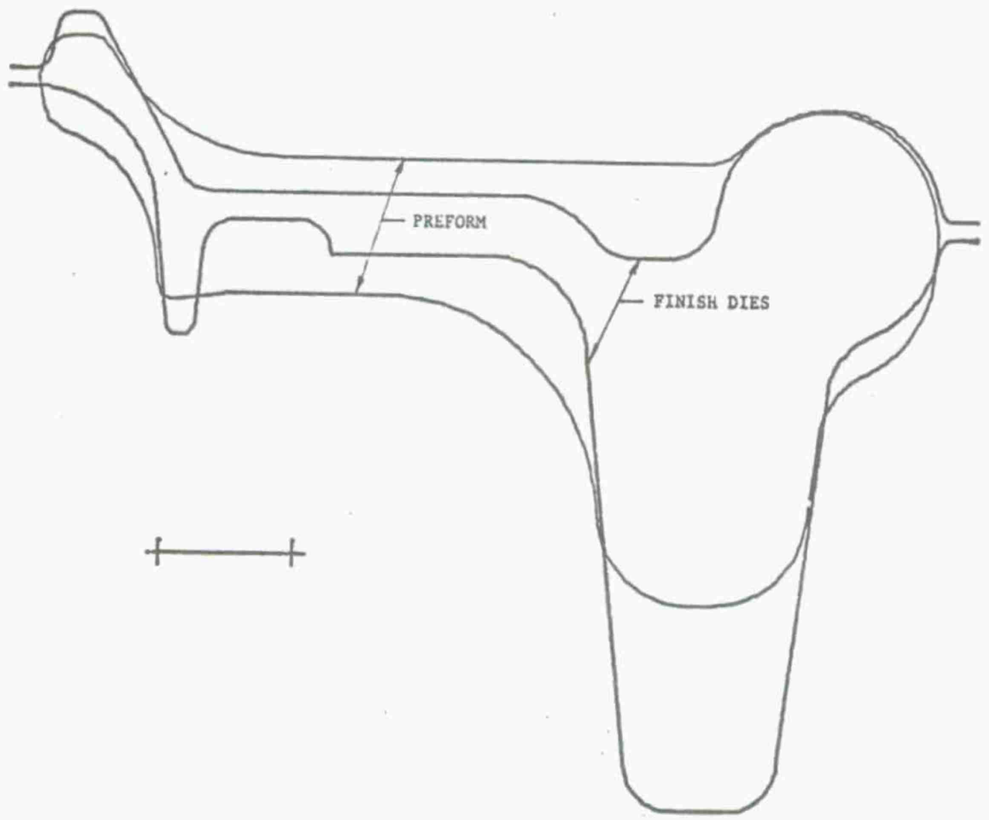


FIGURE VII-22. PREFORM OF TRACK SHOE SECTION WITH FINISHING DIE PROFILE

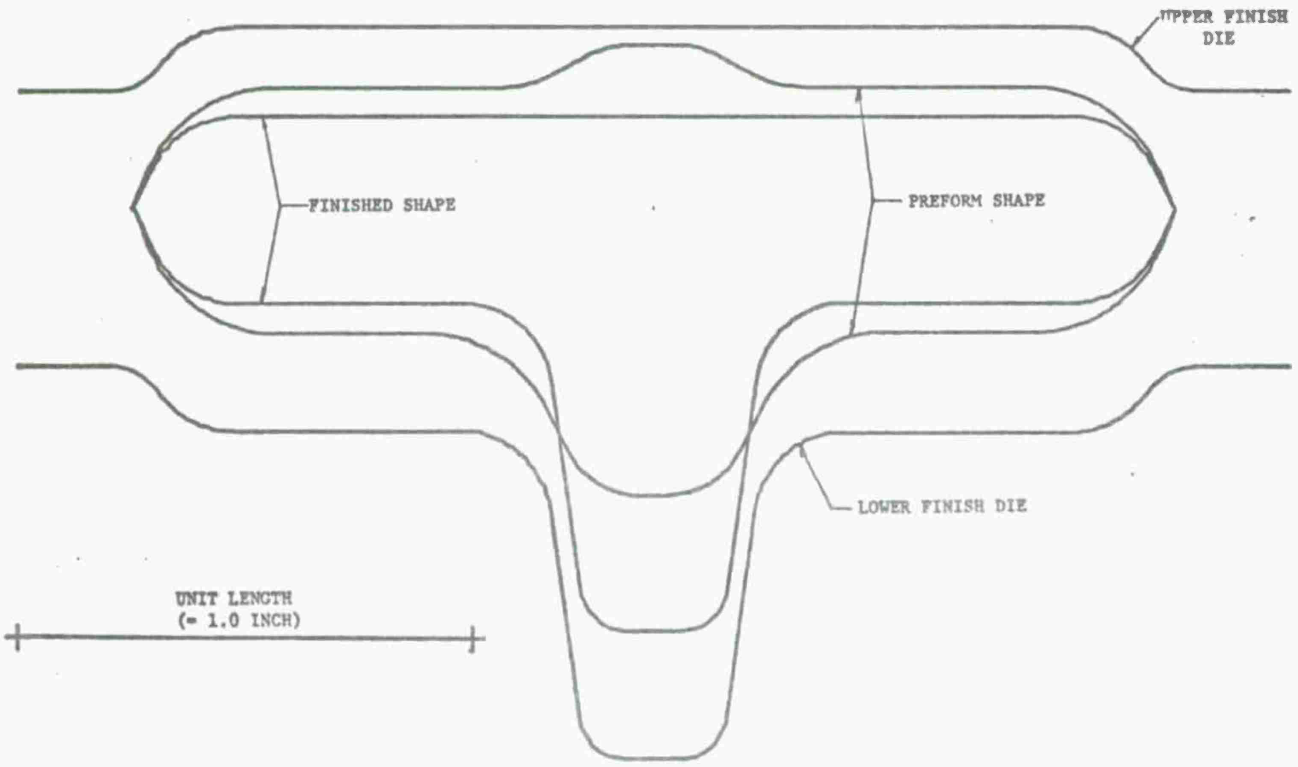


FIGURE VII-23. PREFORM AND FINISH SECTIONS FOR SIMPLE "T" SECTION

Analysis of stress is described in detail in Appendix III. PRLFLW is called from NERGY.

Subroutine PRPROS

This subroutine controls the calling of a number of lower-level subroutines which read and process section data up to the point where stress or preform analysis is to be undertaken. It is the first subroutine called when TRKFRG is initially started. It is subsequently called whenever new section data is desired, or the current section is to be analyzed with different flash parameters.

PRPROS first terminates the graphics mode if the graphics mode is in use, and initializes the flash parameters to their default values. It then requests the designer to enter a value for each flash parameter. If no value is entered, the default value is retained. The question is also asked as to whether or not details of subsequent computations are copied to the print disk file. A "Y" response will cause them to be copied to the file. Any other response will suppress this operation. The program then branches, dependent on whether a new section is to be handled, or the current section is to be used with new flash geometry. If new data is to be used, subroutine RDIOF4 is used to find and read the data from the disk file. If this returns with an error condition, PRPROS returns to TRKFRG. If no errors were found, subroutine ROTATE is called to rotate the section data into a plane parallel to the X-Z plane. Next, if the section is axisymmetric with double flash, the first coordinate point is saved as the coordinate of the axis of symmetry, and all the remaining points are shifted down one position in the part coordinate file.

The area, perimeter, and centroid are found by calls to XSECA. As the last step, the part and flash volumes, and plan area are found and printed via a call to PARAMS. This latter routine is executed for both new sections and existing sections with new flash. Control is then returned to TRKFRG.

Data is passed to PRPROS via the Labeled Common blocks SECTN, SYSPR, and DISPLA.

Subroutine PRTSRF

PRTSRF is the module which creates the model of the preform surface. It does this by generating a mesh of points from the planes created in SEQSRF. The points are offset from the surface by the cutter size specified by the user. Thus, when displayed on the CRT, the surface represents that described by the motion of the center of a ball-end mill. When machined on an NC milling machine, the surface generated will be the true surface of the preform.

PRTSRF is called from the main program, TRACK, after SEQSRF is completed. SEQSRF puts the X-Z planes describing the preform model into sequential order and adds flash planes as specified by the user. When started, PRTSRF reads the file-text header of the scratch file generated by SEQSRF and copies this to the NC output file. The scratch file data header is then read to obtain the number of planes to be processed and the extrema values for the three axis. A scratch direct access file for use by PRTSRF is also created at this time.

The user is then asked to enter the resolution desired. Resolution is taken to be the distance between adjacent machining paths in the Y-Z plane. However, the resolution cannot be less than the larger of the range in X or the range in Y divided by 500. This limitation is due to the size of the arrays defined for PRTSRF. Furthermore, the resolution is not allowed to be less than 0.010 inch. This is felt to be the minimum practical limit for the spacing of the machining paths.

The user is also asked to enter values for the initial Z distance away from the part from which the cutter will start, the draft angle, and the radius of the ball-end mill size which is to be used to machine the preform. The initial Z distance is used to generate the first tool motion which is a plunge directly down into the part. This plunge is the distance specified by the user and is assumed to start at the minimum of X and Y.

A DO-loop is then started which runs for the number of planes specified in the file-data header. Within this loop, the number of points and the Y-axis position of each plane is read. If the number of points exceeds 500 (the defined length of the work arrays), an error message is generated and the routine is terminated. If no error, the X-Z coordinate data pairs are read from the scratch file. These data pairs represent the interpolated preform silhouette which was saved during the stress analysis of the CAD phase.

Subroutine CLPTS is then called to calculate the offsets in X and Z necessary to allow the contact point of the ball-end mill to describe the desired contour when the NC mill directs the motion pattern of the center of the ball. The resulting Z-axis offset values are then saved on the scratch direct access file. Because all sections processed have the same extreme values in X as a result of SEQSRF, and the nature by which CLPTS finds the offset values, all sections are divided into the same number of points. This means the X-axis values resulting from CLPTS are identical for all sections. The Y-axis values for each section are read from the data file. Thus, the data structure resulting from this DO-loop can be characterized as shown in Figure VII-24. The X-axis offset data and the Y-axis position data for the sections are maintained as linear arrays in core memory. The Z-offset data are maintained on the direct access, scratch disk file.

Due to the number of arrays needed by PRTSRF and the amount of core memory available, the preform model can be defined by no more than a 500 x 500 mesh. However, this allows a total of 250,000 unique points to be used if desired! The preform model cut as part of this study was approximately 10 x 10-inches square. Thus, the array sizes would have permitted points to be uniformly spaced 0.020-inch apart. This is far more resolution than justified, and the number of coordinate triplets (X,Y,Z) actually used is estimated at less than 10,000.

After processing all the sections specified and calculating the X and Z offsets, a second DO-loop is started. This loop processes the data in the Y-Z plane. The loop starts by reading back into memory the Z offset data for the Y-Z plane being processed. For each of these planes, X will be constant. The Z data are obtained by using the direct access nature of the file to read the data in increments of the number of X sections. That is, for the first Y-Z plane, the Z data read is

$$Z_1,\ Z_{N+1},\ Z_{2N+1},\ Z_{3N+1},\ \ldots\ Z_{(M-1)N+1}\quad .$$

For the second Y-Z plane, it is

$$Z_2,\ Z_{N+2},\ Z_{2N+2},\ Z_{3N+2}\ \ldots\ Z_{(M-1)N+2}\quad .$$

For the last plane, it is

$$Z_N,\ Z_{2N},\ Z_{3N},\ \ldots\ Z_{mn}\quad .$$

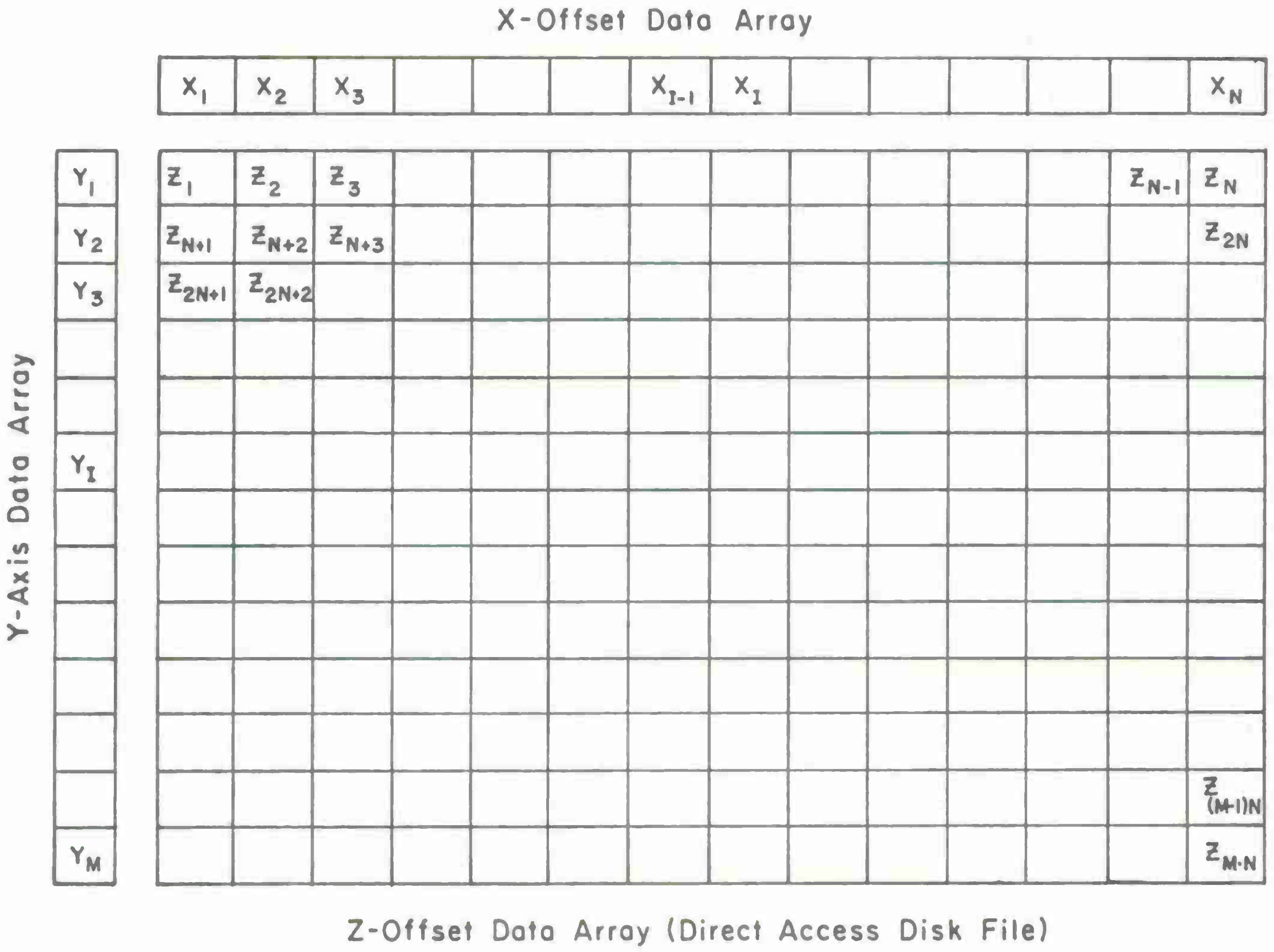


FIGURE VII-24. REPRESENTATION OF DATA STORAGE IN SUBROUTINE PRTSRF

2N+1

$X_1 = X_{min}$; $X_N = X_{max}$

$DX = X_I - X_{I-1} = (X_{max} - X_{min})/N$

$Y_1 = Y_{min}$; $Y_m = Y_{max}$

Y_I = Value Specified for Section

Each Y-Z plane is read, processed to completion, and written to the NC output file before the next Y-Z plane is read.

The first step in processing the data in the Y-Z plane is to eliminate extraneous data points. Unneeded points are likely to occur at the extremes in X where adjacent sections may have identical Z-coordinates. REDUCE is used to find all occurrences of three or more adjacent points which have the same Z value. When such a condition is found, only the first and last point are retained.

The next step in processing is to shift the Y-axis values to provide the specified slope. Referring to Figure VII-25, the slop of each line segment defined by the Y-Z data is found as

$$\alpha = \arctan\left((Z_I - Z_{I-1})/(X_I - X_{I-1})\right) \tag{VII-27}$$

then, let

$$\gamma = \pi/2 - |\alpha| \quad , \tag{VII-28}$$

and θ = specified draft.

Then, if $\gamma \geq \theta$, no adjustment is necessary. For example, in Figure VII-25, the slope of segment 1-2 is zero. Therefore, $\gamma = \pi/2$, and if θ was specified to be 7 degrees, no adjustment would be made.

However, if $\gamma < \theta$, an attempt is made to shift the appropriate points. Segment 2-3 has a slope of $- \pi/2$. Therefore, $\gamma < \theta$. When adjustment is necessary, the line segment is rotated about its midpoint so that its new slope is equal to the specified draft. The new points on the polygon, (i.e., 2' and 3') are determined by finding the equation of the line through the midpoint at the specified draft, and the equations of the line segments adjacent to the segment being shifted using LINEQ. Knowing the equations of the segments, the intersection is found from INT2LN.

The new points calculated (2' and 3') must fit the constraint that $Y_{2'} > Y_1$ and $Y_{3'} < Y_4$.

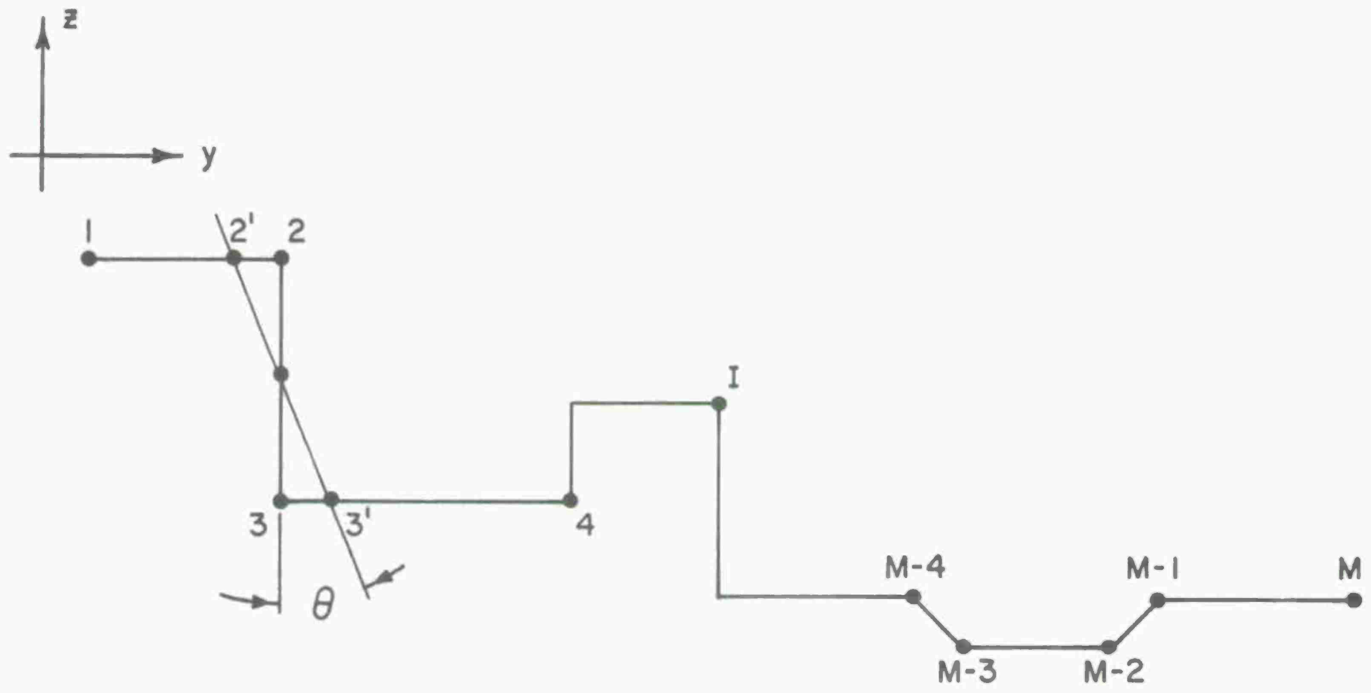


FIGURE VII-25. SHIFTING Y-Z POLYGON TO GENERATE DRAFT SPECIFIED

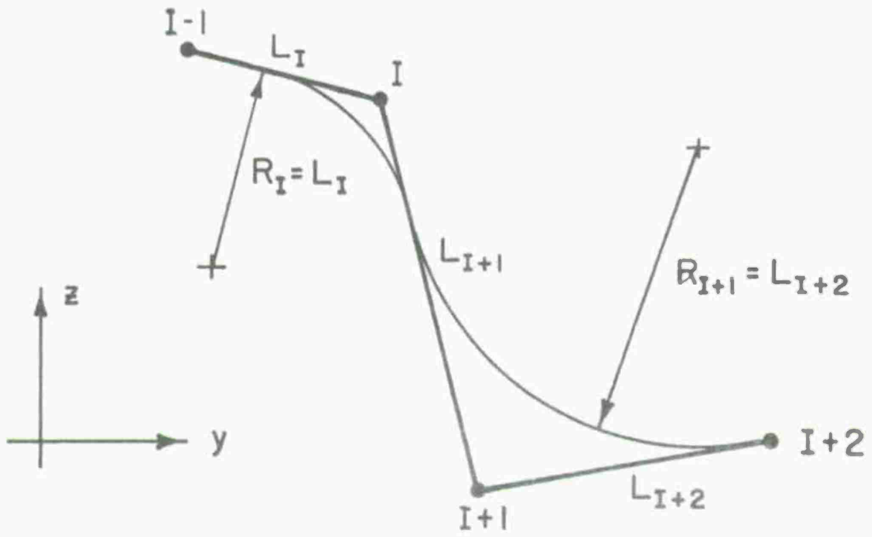


FIGURE VII-26. ESTIMATING THE RADIUS FOR EACH POINT ON A Y-Z POLYGON (Subroutine PRTSRF)

If they cannot meet this condition, an error message is generated and PRTSRF terminates. If this occurs, the user can try repeating the process with the same Y data for the planes, but with less draft. He could also specify the planes to be further apart during SEQSRF. In practice, using 7-degree draft, this situation has never occurred, even when processing the large "horn" projection on the T130 preform.

After shifting the points as required to generate the draft specified, the offset in the Y axis is found using YOFFST. Prior to calling YOFFST, the data has already been offset for the cutter size in X and Z using CLPTS, so YOFFST provides the offset adjustment for Y only. DELETE is then used to eliminate any data points which overlap in the Y direction.

From the Y-Z polygon resulting from the slope shift and offset processing, the next step is to calculate and interpolate radii at each point of the polygon. Referring to Figure VII-26, the length of each segment adjacent to a point is found, and a trial radius for the point is equated to the shorter of the two lengths. For Point I, Radius I is equated to Length I; for Point (I+1), Radius (I+1) is equated to Length (I+2). The radii for all the points on a Y-Z section are calculated first. They are then reduced, as necessary, to eliminate cusps using DECUSP.

The algorithm to use the shorter length adjacent to a point as the radius is an arbitrary technique. This technique was decided upon while discussing ways in which a slab of one profile could be smoothly blended into the adjacent slab. The tests made in which preform models were machined using this method indicate it works reasonably well. If any changes are to be made in sizing the radius, it is felt the change should be in the direction of enlarging the radius. Alternative strategies include picking the larger of the two lengths, or the average length. Whatever radius value is selected for the first estimate, it will always be reduced by DECUSP, if it is too large.

After estimating and decusping, as required, the radii are further modified by considering the cutter size. This procedure is described in detail in Appendix IV. The Y-Z polygon with the specified draft and calculated radii is then interpolated using FITARC. This changes the radii to a series of short line segments. The coordinates of these line segments serve as positioning data in the NC output tape. The sequence of every second section is then reversed to generate the back and forth motion along the Y-axis. That is, passes 1,3,5,7 . . . are made moving in the +Y direction, and passes 2,4,6,8 . . . are made moving in the -Y direction. Variable IDR is the direction pointer and its sign is reversed after each Y-Z section is processed.

When all Y-Z sections are completed, a Z-axis move away from the part is made. This is the same size as the initial Z-axis plunge into the part, but in the opposite direction. Trailer code is then punched, and the output file terminated.

A number of additional points should be noted concerning the use and operation of PRTSRF. When the X-Z plane processing is completed, the message "X-Z PROCESSING COMPLETED" is generated. Also, at the end of processing every tenth Y-Z plane, the message "PATH XX COMPLETED" is output where XX = 10,20,30 . . . Because of the amount of processing done in PRTSRF, these messages are used to assure the operator that the program is running correctly.

If PRTSRF is compiled with the D option, tool paths for several different sizes of ball mill may be generated, without having to re-run SEQSRF for each pass. The user will remain in PRTSRF as long as he specifies a nonzero, integer value in response to the queary "Y-Z PASS TO LIST" from the program. This information request is made only if the compile D option is used.

No data is passed to PRTSRF in the calling sequence. Labeled Common blocks SYSPR and SECTN are used for data. Labeled Common blocks DISPLY, STRSPT, DEFEL, and the Unlabeled Common block are used for array storage, since these blocks of core would otherwise be unused. A new Labeled Common block, PUNCHO, is used for data transfer between PRTSRF and the group of subroutines, PNCHNC, which actually generates the NC output tape data file.

The results of PRTSRF are shown in Figures VII-27, and VII-28. Figure VII-27 is a copy of the CRT display of the cutter paths using the following specifications:

(1) Resolution between Y-Z passes: 0.100 inch

(2) Draft angle 7 degrees

(3) Cutter radius: 0.25 inch

(4) Initial Z distance from part: 2 inches.

This figure is obtained using NCDATA, described in Appendix IX.

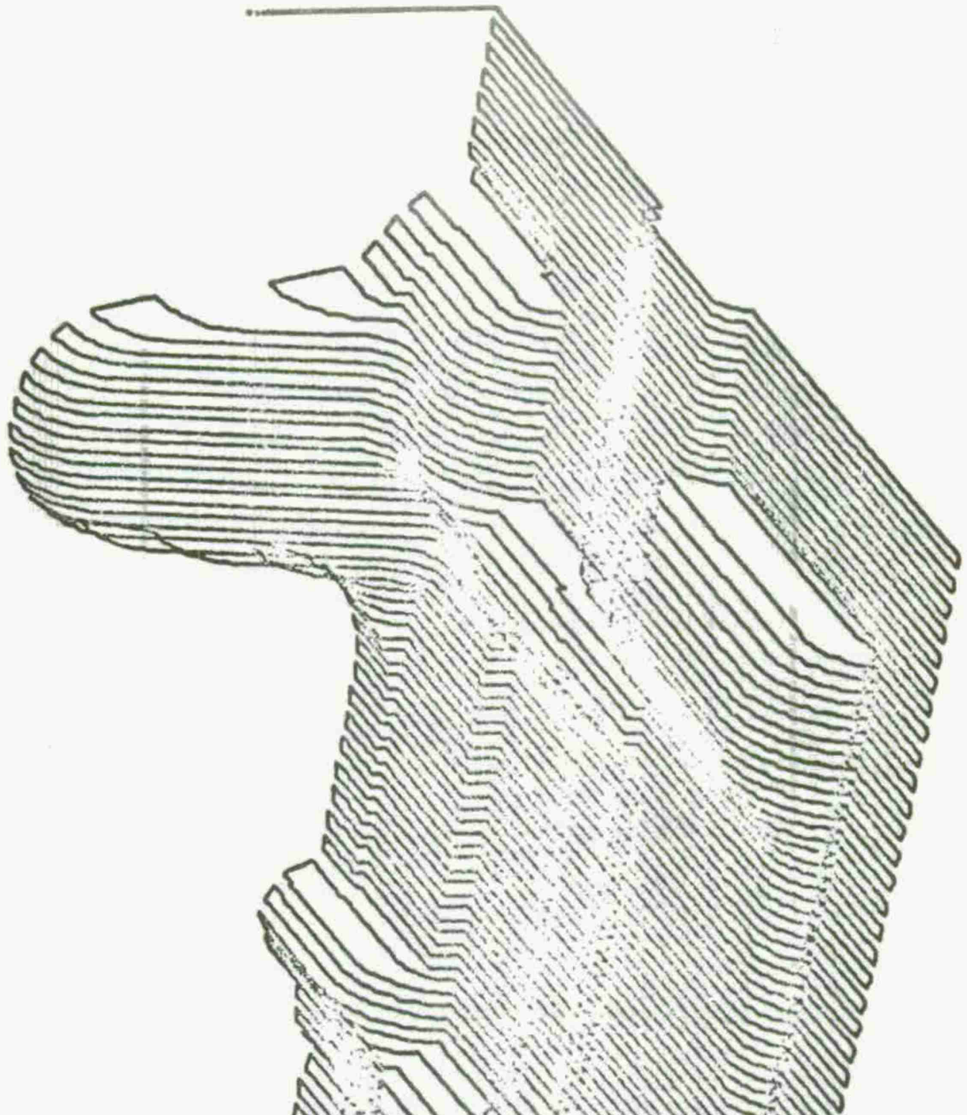


FIGURE VII-27. PLOT OF CUTTER PATHS GENERATED BY PROGRAM NCDATA

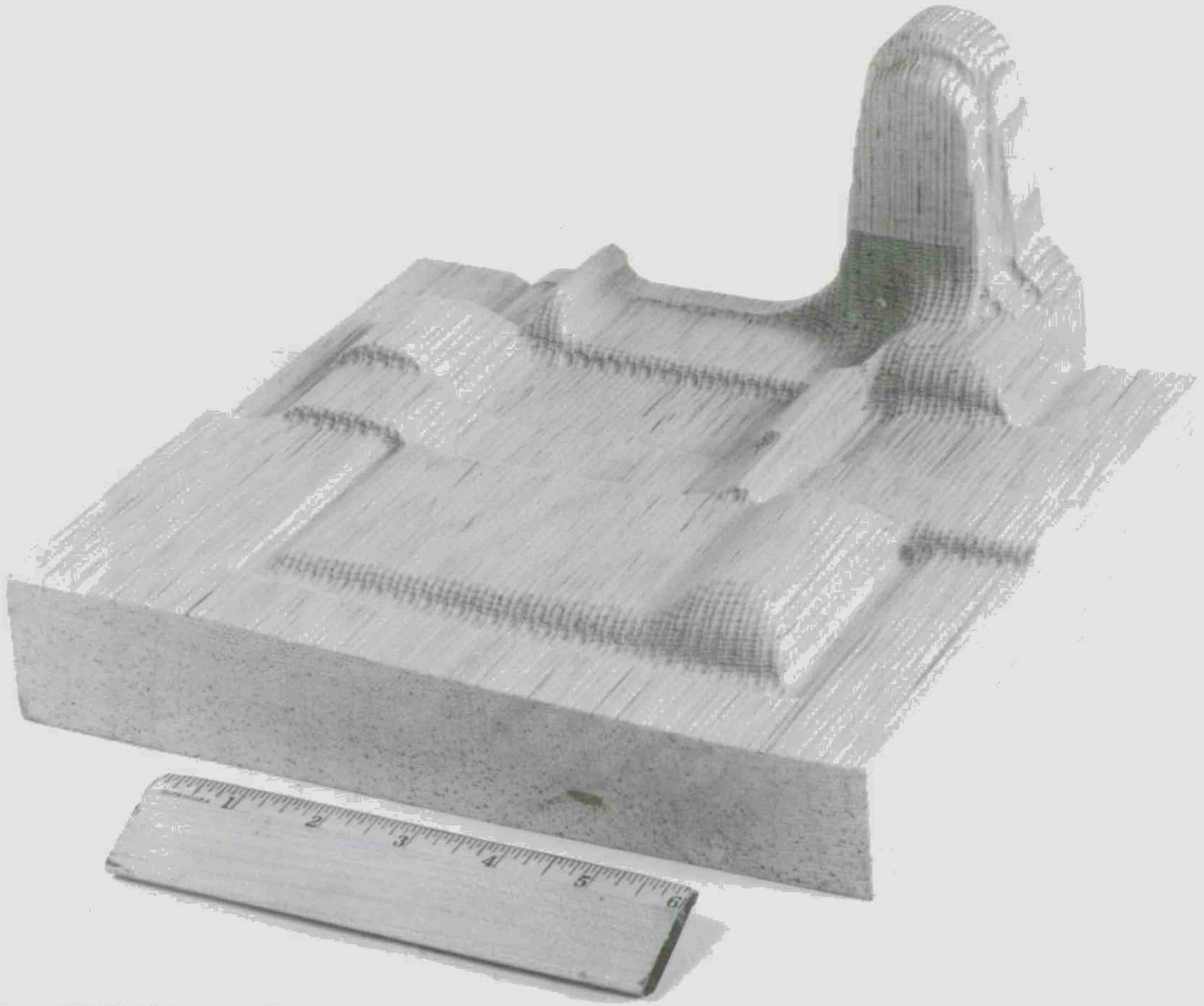


FIGURE VII-28. WOOD MODEL OF UPPER PREFORM DIE MACHINED ON BATTELLE CNC MILLING MACHINE

Figure VII-28 is a photograph of a wood model of the T130 preform cut on the BCL CNC milling machine using the parameters specified above.

Subroutine PXYZ(INC,NAME)

PXYZ takes the integer data representing incremental motion commands for any of the three axes, converts the data to individual ASCIII or EIA characters, and then uses NCOUT to output the characters. The routine operates by repeatedly dividing the input value, INC, by successively smaller powers of 10. The quotient, if not a leading zero, is output via NCOUT. The quotient is next multiplied by the same power of 10 and subtracted from the initial value to get the next value.

For example, let INC = 1234.

Since the largest integer value which can be expressed is ±32768, the first power of 10 is 10^4 or 10000. Thus, the first digit to be output is

$$\text{INT} = \text{integer}\ (1234/10000) = 0 \quad . \tag{VII-29}$$

$$\begin{aligned}\text{INC}_{\text{New}} &= \text{INC}_{\text{Old}} - \text{INT} * 10000\\ &= 1234 - 0 * 10000\\ &= 1234 \quad .\end{aligned} \tag{VII-30}$$

On the next pass, the power of 10 is reduced to 10^3 or 1000. Then,

$$\text{INT} = \text{integer}\ (1234/1000) = 1 \quad . \tag{VII-31}$$

The ASCII representation of 1 (61_8) is sent to the output file and the new value becomes:

$$\begin{aligned}\text{INC} &= 1234 - 1 * 1000\\ &= 234 \quad .\end{aligned} \tag{VII-32}$$

This process repeats until all digits have been processed.

The variables in the calling sequence are as follows:

INC: The integer value for the distance to be moved along the axis NAME

NAME: The ASCII code for the axis along which a move of INC size is to be made.

Subroutine RADEXP(NRIB)

RADEXP is used to expand all radii, except the corner radii of ribs. Each point on the preform polygon is tested to determine if it is the first corner of a rib. If it is, this point and the next point (the second corner of the rib) are skipped. If the point being tested is not the corner of a rib, the radius of this point is expanded by the functions:

$$R' = R\ (1 + e^{-3|R|})\quad . \qquad \text{(VII-33)}$$

R'/R as a function of R is shown in Figure VII-18.

The calling sequence argument NRIB is the count of the number of ribs on the preform. Data is also passed via Labeled Common blocks DEFEL and SYSPR.

Subroutine RDIOF4

The function of RDIOF4 is to read, from a card-image file, input section parameters and the coordinate data defining a particular cross section. The designer specifies the data file to use as part of the start-up sequence dialog. The data is formated as follows:

(1) The very first card contains the part name, number, or any other desired alphanumeric information. It is read and written in A format, and no operations are performed on the data.

(2) Following the title card is the section header card. This contains (a) the section number, (b) a value indicating the type of section (i.e., plane or axisymmetric), (c) the depth or wedge angle of section, and (d) the maximum and minimum values of the finished section, if this section represents a preform.

(3) After the section header card comes the coordinate data cards, 1 card for each coordinate point on the cross section.

(4) Between each set of cards defining a particular section, (i.e., between the last coordinate data card of one section and the header card of the next section), a card with a "9" in column 45 is used to indicate the end of the section.

The subroutine operates in the following manner:

The number of coordinate points read is initialized to 1, and the section header is then read. If any errors occur on the read, the error flag is set to 3 and control is returned to TRKFRG. If the section header is read correctly, the data points are read until the end-of-section card is encountered. The index for the number of points in the section is incremented once after each successful read. When the end-of-section card is found, the index is decremented once and then a test is made to see if more than two points have been read. (Three points are needed to define a plane). If this test fails (less than 3 points), the error flag is set to 5 and the routine returns to the calling program.

If more than 50 sets of data are read before the end-of-section card is found, the error flag is set to 4. The program continues to read, however, until the end-of-section card is found, at which time a return is made. This is an attempt to get the data back into synchronization by reading up to the next section header card.

If the section read was not the section specified, RDIOF4 loops and reads the next section. This cycle will continue until the proper section is found, or until all data is read. When the proper section has been found and its data read, a test is made to determine if there are more than 25 data points on the upper surface. The program is limited to polygons with 50 or fewer total data points, and 25 or fewer data points on either surface. If there are more than 25 points on the upper surface, the error code is set to 6.

The format of the data cards is as follows:

(1) Part title: cc 1-80, any alphanumeric data. Read as 20A4.

(2) Section header:

(a) cc 1-5, Section number, I5.

(b) cc 6-9, blank.

(c) cc 10, section type: 1 = plane strain, 2 = axisymmetric single flash, 3 = axisymmetric double flash (I1).

(d) cc 11-20, section depth if plane strain, wedge angle if axisymmetric. F10.4.

(e) cc 21-30, blank or the maximum Z value for this same section when it was previously analyzed. F10.4.

(f) cc 31-40, blank or the minimum Z value previously found. F10.4.

(g) cc 41-50, blank on the maximum X value previously found. F10.4.

(h) cc 51-60, blank or the minimum X value previously found. F10.4.

(3) Coordinate data: X,Y,Z and R as 4F10.4. R is positive if a fillet radius and negative if a corner radius.

(4) End of section: 9 in column 45.

Data is passed to and from this subroutine through the Labeled Common blocks SECTN and SYSPR.

Subroutine REDUCE(X,Y,N)

REDUCE is used to eliminate superfluous points from an array. Such instances occur when there are three or more adjacent points with the same Y value. Thus, REDUCE only eliminates extra points if they lie on a common horizontal. It does not check for co-linear points which are not horizontal. Its function is to reduce, if possible, the number of Y, Z coordinates processed by PRTSRF. In several of the cases studied, it was found that the Y-Z tool paths at the extremes in X were essentially horizontal lines which could be described by fewer coordinates than originally contained in the arrays.

REDUCE operates using a DO-loop index (J), a second programmed index (K), and a flag. If two adjacent points do not describe a horizontal line, K is incremented, the flag reset to zero, and the current point copied to the output. If the two points are horizontal and it is the first occurrence of this condition, K is incremented but the flag is set to 1 before copying the output point. If the two points are horizontal and the flag is set, it implies the current point plus at least the previous two are horizontal. When this occurs, K is not incremented, so when the current point is copied to output, it replaces the previous last point.

The following parameters in the calling sequence are as follows:

X,Y:	Arrays for coordinate data (input and output)
N:	Number of elements in X and Y (input and output).

Subroutine RESLTS

This subroutine outputs the summary results to the terminal and print file for all sections analyzed. The values reported include:

(1) Total plan area
(2) Total finished part volume
(3) Total volume (part volume plus flash volume)
(4) Total vertical load
(5) Center of load - X and Y coordinates.

This subroutine is called from TRACK as a result of the operator indicating he has completed the design of all sections desired.

Data is passed to the routine via Labeled Common blocks SYSPR and SECTN.

Subroutine RIBFIL (X1, Z1, X2, Z2, DZ)

Given that a rib fillet point is to be moved vertically by an amount DZ, this subroutine finds the new fillet coordinates such that the slope of the rib flank remains the same. The new fillet Z value is first found by adding DZ to the original fillet Z value. The X coordinate of the new fillet point is then found by calling function XYINTR and passing to it the original coordinates of the rib flank plus the new fillet Z value.

The variables in the calling sequence are as follows:

(1) X1, Z1 - The coordinates of the fillet of a rib (input and returned).

(2) X2, Z2 - The coordinates of the rib corner adjacent to fillet X1, Y1.

(3) DZ - The vertical amount the fillet is to be moved.

Subroutine RIBID (ITYPE,XPF,ZPF,RPF,NPF,IRIB,NRIB)

This subroutine locates the first corner of all ribs on a profile. It does this by scanning the profile for all occurrences of two consecutive corners (negative radii). When each such occurrence is found, the height to width ratio of this feature is found by subroutine RIBPRM. If this ratio is less than 0.3, the feature is considered to be a raised web rather than a rib. If the height to width ratio is greater than 0.3, the slopes of the lines between each corner and its adjacent fillet are found. If these are of opposite sign (one positive and one negative slope), the position is considered to be a rib.

The parameters in the calling sequence are:

(1) ITYPE - The type of section (plane or axisymmetric) represented by the data (input).

(2) XPF,ZPF,RPF - Arrays defining the coordinates of the profile (input).

(3) NPF - The number of coordinate points on the profile (input).

(4) IRIB - An array used to store the index to the coordinate of the first corner of each rib (output).

(5) NRIB - The number of ribs on the profile (output).

Subroutine RIBPRM (J,RHS,RWB,RWT,RIBRAT,XS,XT,RH2)

This subroutine identifies a number of significant parameters for a rib. The rib is identified by the index, J, to its first corner point. Referring to Figure VII-29, RIBPRM first finds the height (DY1 and DY2) of each side of the rib. The smaller of these two heights is called RHS, and the larger is called RHL. The point of intersection is then found between the horizontal line through the fillet point defining RHS and the opposite side of the rib. This point of intersection is referred to as XT in Figure VII-29. XS is the X-coordinate of the point defining the bottom of the rib ahead of the corner identifying the rib. The horizontal distance between XS and XT is the bottom rib width (RWB). The horizontal distance between the two corners of the rib is the top width (RWT). The rib ratio (RIBRAT) is defined as the ratio of the rib height to the average rib width, or

$$\text{RIBRAT} = 2\text{RHS}/(\text{RWB} + \text{RWT}) \quad . \qquad \text{(VII-34)}$$

This subroutine is used by the subroutines RIBID, ADDRIB, and PREFRM.

In addition to the variables passed in the calling sequence, described above, Labeled Common block DEFEL is used to pass the coordinate arrays and number of coordinates defining the profile.

Subroutine RIBRAD (X1, X2, R1, R2)

Given the X coordinates of the two corners of a rib and the radii of each corner, this subroutine modifies the radii such that the two radii are equal to each other and equal to half the distance between the two corners. The radii are not expanded if greater than a preset limit. This value is currently set to 1.5 inches. This subroutine assumes the line joining the two corners is horizontal, or nearly so.

The variables in the calling sequence are as follows:

(1) X1, X2 - The X-coordinates of the corners of the rib (input).

(2) R1, R2 - The original and expanded rib corner radii (input and output).

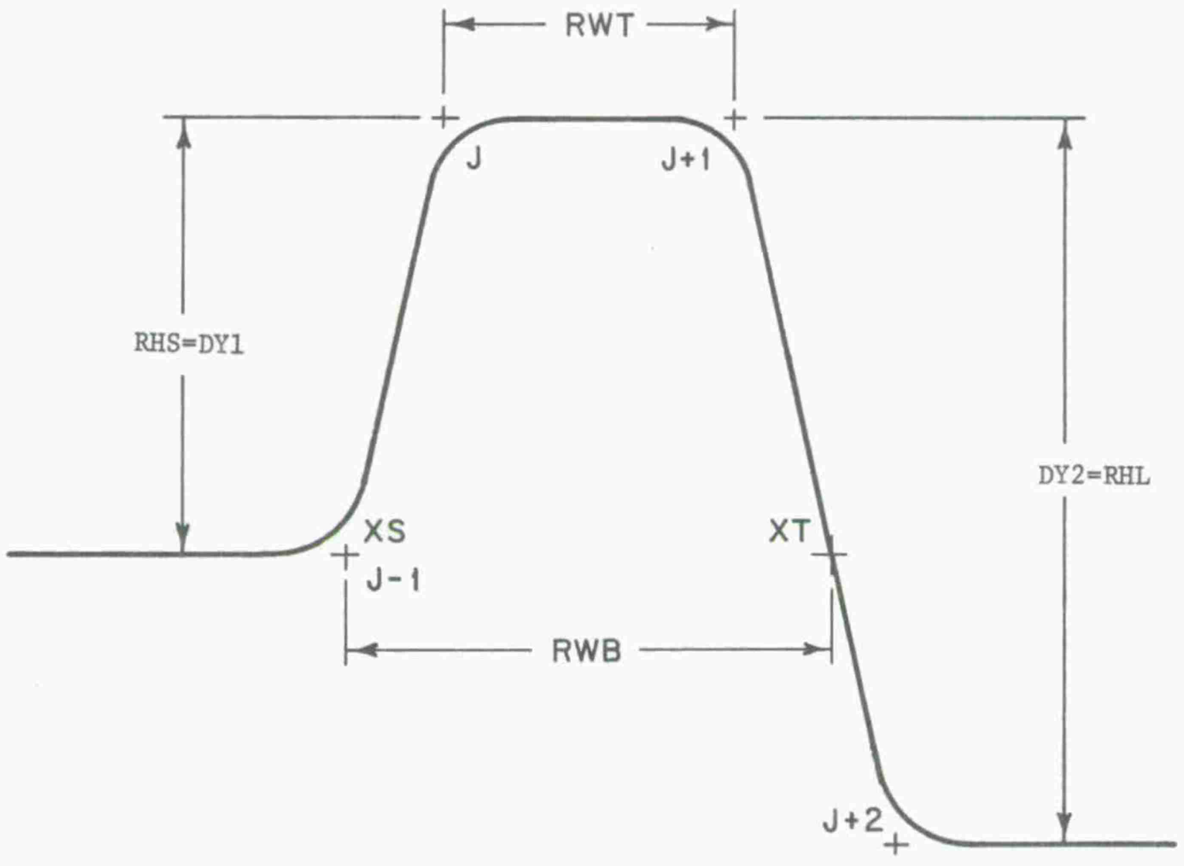


FIGURE VII-29. NOMENCLATURE USED BY SUBROUTINE RIBPRM

Subroutine ROTATE

ROTATE is used to transform a series of coordinates defining a section in any plane parallel to the Z axis to the coordinates defining the same section in a plane parallel to X and Z. The angle of the original plane relative to the X axis is first found as the arc tangent of the line defined by the first and third points. The second point is not used since for an axisymmetric section, the X and Y coordinates of the first and second points can be identical. The sine and cosine of this angle are then found and used to rotate all the X coordinates. Since the plane is rotated parallel to X and Z, all rotated Y values are the same and, therefore, only a single rotated Y value is found.

Data is passed to and from this routine via the Labeled Common blocks SECTN and SYSPR.

Subroutine SAVENC

SAVENC is used to save the upper and lower die profiles on disk. The data representing these profiles may then be used in the CAM phase to develop the cutter paths which will machine the part (EDM electrode) surface. The subroutine is called in response to a user directive issued in subroutine STRESS.

When first called, SAVENC checks for the conditions given below. If any of the conditions are not met, an appropriate message is typed and the routine returns to STRESS.

Conditions Required for Saving Die Profiles for CAM Processing:

(1) Plane strain sections

(2) Section plane parallel to X-Z plane

(3) Section not previously saved

(4) Section fits between the gutter maximum and minimum limits.

If the above conditions are met, SAVENC generates two new points on both ends of the profile. These new points are illustrated in Figure VII-30. A section header is then written to the file. This includes the following:

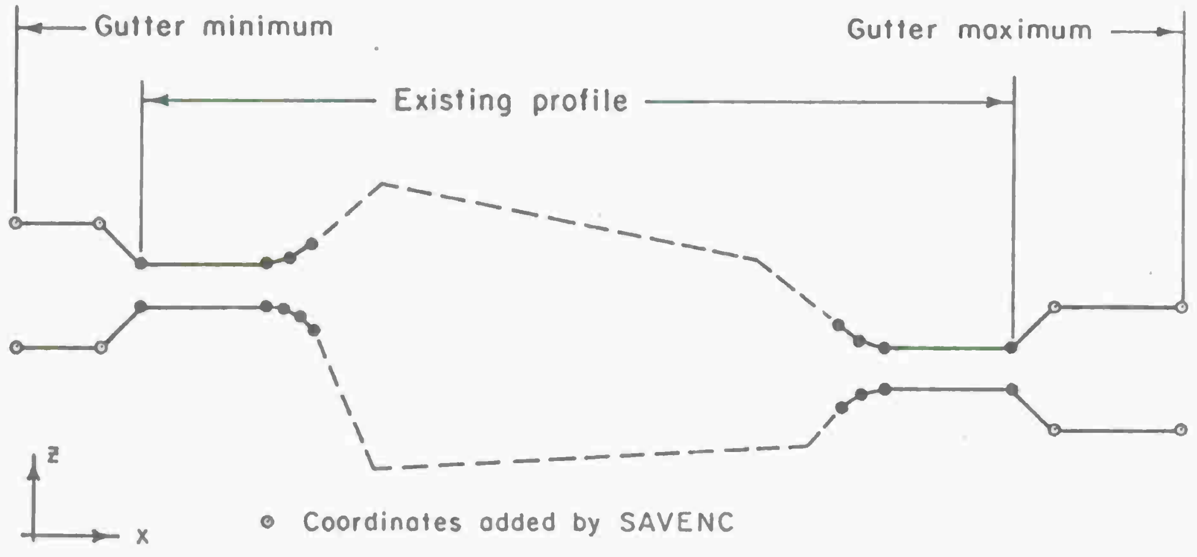


FIGURE VII-30. DIE PROFILE COORDINATES SAVED BY SUBROUTINE SAVENC

- Section ID number
- Section type code (at present, always 1)
- Profile surface indicator (1 = upper surface, 2 = lower surface)
- Number of coordinates written to file.

The X, Z data itself is then written to the file. After the upper profile is written in the file, the lower profile is written in a similar manner. Before returning to STRESS, the logical flag is set to prevent the data from being written to the file a second time. SAVENC uses unformatted (binary) write statements to conserve disk space. Because of this, the file created by SAVENC cannot be listed using the PIP utility program.

No variables are passed in the calling sequence. All data is accessed through blank (unnamed) common, and labeled Common blocks SYSPR, SECTN, and DISPLA.

Subroutine SAVPLY

This subroutine saves the designed preform (blocker) polygon on a file. The format of this file is the same as that of the original finish die polygon file. This allows the preform polygon to be analyzed in the same way the finisher polygon was, in order to add the flash to the preform, and to make stress and load calculations.

The subroutine first tests the state of logical variable SAVBLK. If TRUE, the preform polygon for this section has not been previously saved, and the routine continues. If SAVBLK is FALSE, the message "Preform polygon for this section already saved" is typed and the routine terminates. If originally TRUE, SAVBLK is next set FALSE, and the section header containing the section number, section type, depth or wedge angle, and maximum and minimum X and Y values is written to the file. If an axisymmetric, double flash section, the X coordinate of the offset axis is then inserted into the coordinate arrays as the first value. The polygon coordinates are next rotated back to their original plane and then written to the file.

This subroutine is called from subroutine PREFRM as a result of an operator directive.

No variables are passed in the calling sequence. All variables are accessed through the Labeled Common blocks SYSPR, SECTN, and DEFEL.

Subroutine SCALZZ (X,Y,IX,IY,J,K)

SCALZZ converts coordinate points in the user's real units to the raster coordinates needed by the CRT. The scale factor used is that calculated by subroutine INITDS. An offset is added to the Y values to prevent the image from appearing at the bottom of the screen. The bottom of the CRT is reserved for the text scroller.

The passed parameters are:

(1) X,Y - Arrays of real values in the user's subject space (input).

(2) IX, IY - Arrays of integer values for the coordinates converted to the CRT object space (input).

(3) J,K - The first and last point in the arrays which are to be scaled (input).

Data is also passed via Labeled Common block DISPLA.

Subroutine SEQSRF

SEQSRF is the first subroutine called when the user indicates the CAM phase is to be entered. This routine uses the data saved on the NC Prep file by subroutine SAVENC. The purpose of SEQSRF is to create a solid, three-dimensional model of the preform part from the two-dimensional planes describing the preform. This is done by specifying two Y positions for each profile plane, thus creating a slab from each plane. By specifying the sequence of the sections and their Y-axis positions, the solid preform is created as a series of solid slabs.

When called, SEQSRF first asks for the name of the input data file, opens this file and two temporary scratch files. It then asks the user to indicate which surface (upper or lower) is to be processed. If no entry is made, the upper surface is assumed. The data file is then rewound and the file header read. This positions the file so that the section header for the upper profile of the first section in the file will be the next item to be read. The user is then asked to enter the ID number for the section desired, and its Y-axis position. The Y-axis values are tested to ensure that they are in increasing order. The data file is then read until a match is found on both the section number and the surface indicator (upper or lower). If the entire file is read without finding a match, a message to this effect is typed.

When a match is found, SEQSRF next checks if this is the very first section processed. If it is, the user is asked if flash is desired at this end. If flash is desired at the first Y plane, four new planes are created based on the first and last four coordinates of the first section specified. The planes are located along the Y-axis based on the flash dimensions.

Given the first section plane at Y-axis position YP, and whose first four coordinates are (X1,Z1), (X2,Z2), and (X4,Z4), the coordinates and positions of the four flash planes are as follows:

	X, Z Coordinate Pairs			
Y Position	1	2	3	4
YF1 = YP - FR	(X1,Z1)	(X2,Z2)	(X3,Z3)	(X4,Z4)
YF2 = YP - FR - FW	(X1,Z1)	(X2,Z2)	(X3,Z3)	(X4,Z4)
YF3 = YP - 2FR - FW	(X1,Z1)	(X4,Z1)		
YF4 = YP - 2FR - FW - 1.	(X1,Z1)	(X4,Z1)		

where FR = Flash radius
FW = Flash width.

Similar sets of coordinates are generated for the right-hand side of the flash planes, using the last four coordinate pairs of the first section plane. A straight-line is assumed between the innermost pair of points of each flash plane. The coordinates and position of the Y-axis flash planes is shown in Figure VII-31. After SEQSRF generates each flash plane, it writes the data to the first scratch file. Before writing the X, Z coordinate data, the number of points on the section and the Y-axis position of the section are written to the file.

After generating the flash planes, or skipping this process if requested by the user, the Z-axis values of the section plane are tested for new maximum and minimum. The first and last X-axis coordinates are tested for new extrema in this direction, followed by a test of the Y position. The purpose of finding the extrema in the three axes is to establish the size of a block into which the solid preform model will fit. Such a block would

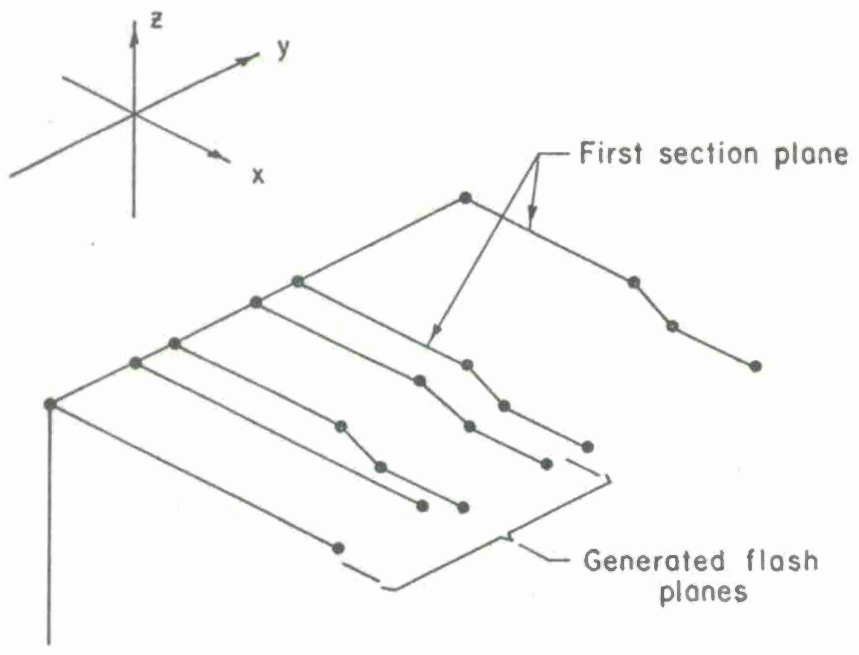


FIGURE VII-31. FLASH PLANE GEOMETRY GENERATED BY SEQSRF

be the minimum size piece of graphite from which an EDM electrode could be machined. Once the extrema are tested for, the section data itself is written to the first scratch file.

The program then loops and asks for the next section number. Each time the user is asked to enter the section number and Y-axis position, the previous value for each of these is shown. If the user wants to use this previous value again, he only types Return. This saves time in that usually a section plane is specified twice to form a slab, and a Y position is specified twice at the transition from one section to another. A Return is interpreted as a zero. Thus, if a Y position value of zero is desired, it must be approximated by entering a very small value (i.e., 0.0001). When all sections have been processed, the user enters -999 in response to the section number query in order to break out of the loop. When the user indicates he has processed all sections desired, he is asked if flash is desired on the plus Y-axis end of the shape. If the user answers Y (Yes), four additional flash planes are generated in a similar manner to that described above. After the plus Y-axis flash is generated, or after this step is skipped, the part title, the number of sections processed, and the maximum and minimum in the three axes are written to the second scratch file. The first scratch file is then rewound and copied to the second scratch file. The end of the second scratch file is marked and the file rewound, and the first scratch file is closed (terminated). SEQSRF then returns to TRACK which originally called it.

No data is passed to SEQSRF in the calling sequence. The data required is obtained from the data file specified by the user and from Labeled Common blocks SYSPR and SECTN. It should be noted that no graphics displays are used in the CAM processing phase of the system. Therefore, Labeled Common block DISPLA is used for temporary storage.

Subroutine SETUPS(I)

SETUPS performs the following functions based on the value for I:

If I = 1:

(1) Initialize running sums, such as total volume and total load, to zero. Set other variables to appropriate initial values.

(2) Establish the linkage between the track-shoe programs and the graphics handler.

(3) Create and open new output disk files; open the existing data file.

(4) Establish gutter limits. The gutter limits are the smallest x value for all sections to be processed and the largest x value for all sections. An additional allowance on either end should be provided to allow for flash. The gutter limits are specified in absolute coordinates. The difference between the two limits is the width of the block from which a model of the part could be machined.

If I = 1 or 2, the user is asked to enter the number of the section he wants to process. If no value is entered, the next section in the data file will be used. If a negative value is entered, the data file is rewound and the file header read so the file is positioned at the section header for the first section. A number of variables which must be initialized each time a new section is processed are also set as required.

If I = 1, 2, or 3, the user is asked to enter values for the flash parameters. The values entered will remain in effect until the user again changes them.

SETUPS is called only from the main program TRACK.

Data is passed via Named Common blocks SYSPR, SECTN, and STRSPT.

Subroutine SHRPTS (X1,Y1,X2,Y2,X3,Y3,X4,Y4)

This subroutine provides an approximation to the shape of the shear surface between points (X1,Y1) and (X2,Y2) of the cavity of the forging.

Initially, the direction of flow is not known. The procedure, therefore, attempts to determine the shape of the shear surface such that the vertical stress midway between the points (x_1,y_1) and (x_2,y_2) is minimum. As shown in Figure VII-32, assuming the simple rectangular cavity model with L as the length and t_1 as the thickness, the height H_{S_1} is determined using the relation:

$$\frac{H_{S_1}}{t_1} = 0.8\ (L/t_1)^{.92} \qquad \text{for } L/t_1 > 2 \qquad \text{(VII-35)}$$

$$= 1 \qquad \text{for } L/t_1 \leq 2 \quad . \qquad \text{(VII-36)}$$

This height approximately makes the stress minimum at the center, provided the direction of flow is from left to right or towards the middle. Similarly, the height H_{S_2} is determined. In general, the two surfaces will not meet at the center. A sloped shear surface is therefore assumed between points S_1 and S_2.

The variables in the calling sequence are as follows:

(1) X1,Y1 - Coordinates of left-hand cavity boundary (input).

(2) X2,Y2 - Coordinates of the right-hand cavity boundary (input).

(3) X3,Y3 - As input, the coordinates of the point on the opposite die surface, directly above or below the left-hand cavity boundary. As output, the coordinates of the left-hand point found on the shear surface (S_1).

(4) X4,Y4 - As input, the coordinates of the point on the opposite die surface, directly above or below the right-hand cavity boundary. As output, the coordinates of the right-hand point found on the shear surface (S_2).

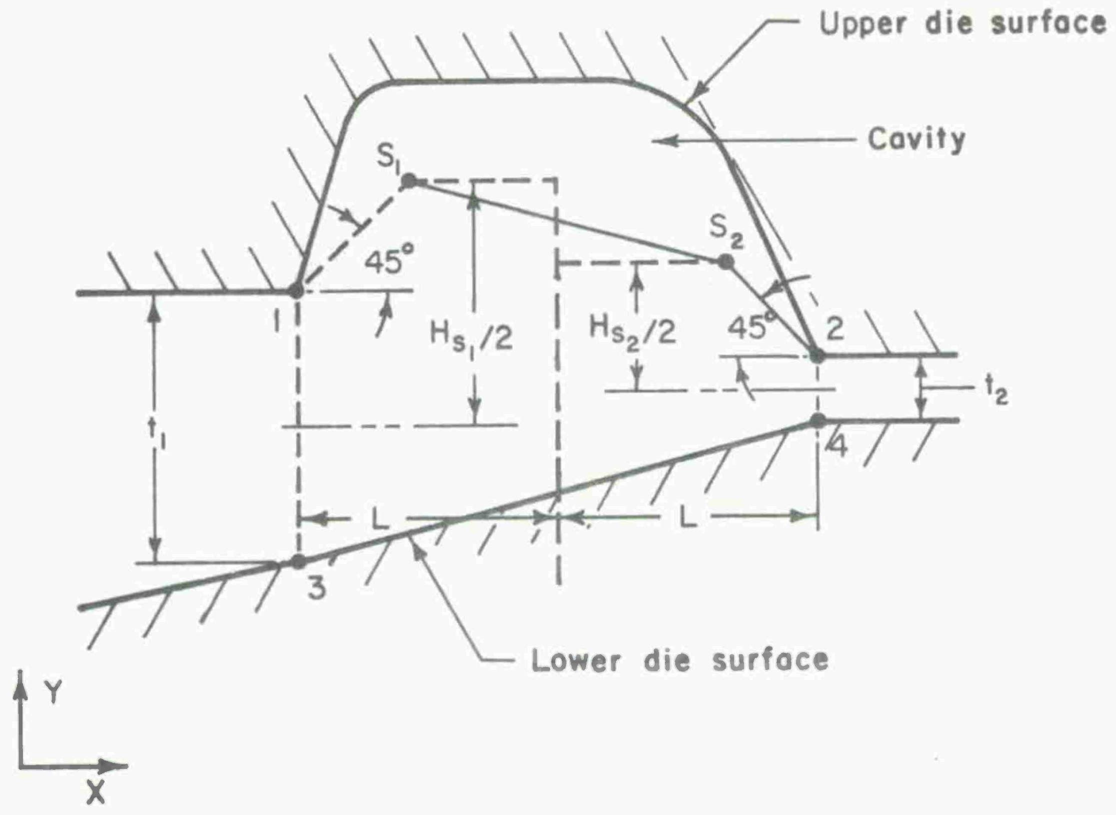


FIGURE VII-32. APPROXIMATE CREATION OF A SHEAR SURFACE IN A CAVITY

Subroutine STRESS

Using the trapezoidal deformation elements found in DEFREL, STRESS finds the stress imposed on each deformation element. For double flash sections, this is done by calling subroutine DBLFST. For single flash sections (always axisymmetric), the stress calculation is done within STRESS as follows:

A logical flag is first set true, indicating that the center of load is to be found for each element. Then the wedge angle of the section is converted to radians, running sums for stress and moment are cleared, and the stress at the extreme right-hand flash boundary is equated to the flow stress of the material. As a generalization, the flow stress can be equated to the yield strength of the material at the forging temperature.

A DO-loop is then entered which calculates the stress on each element, moving from right to left (that is, from the flash boundary to the axis of symmetry. It should be noted that for axisymmetric, single flash sections, the axis of symmetry and the neutral axis are identical). The stress calculated for the left boundary of each element becomes the starting stress on the right boundary of the next element to the left. The running sums for the load and moment are also maintained within the loop. When the loop is completed, the maximum stress, neutral surface, load, and center of load are found.

STRESS then branches, prints the stress distribution if a detailed printout was previously requested, and scales the stress distribution to fit the CRT. The distribution is scaled vertically so as to appear as large as possible. The distribution is then plotted and the results printed for the operator.

The user is then asked if he wishes to save the die profile. If he responds Y (yes) and the section has not been previously saved, SAVENC is called. The upper and lower die profiles saved in this manner may be subsequently accessed in the CAD phase to develop the part or preform surfaces. Once a section is saved, a flag is set to prevent the section from being saved again. After SAVENC is completed, sums for such variables as total plan area, total volume, total load, etc., are updated by adding the values for the current section to the sums.

If the section had flash on both sides, after returning from subroutine DBLFST, a test is made to determine if a new deformation element was generated while finding the neutral axis. If it was, the x-axis and stress value for this element are inserted into their proper position in the stress distribution.

Subroutine STOPNC

STOPNC causes an "MØØ" code to be punched on the NC output tape file. This code indicates a program stop to NC controllers. STOPNC is called at the end of PRTSRF after all coordinate data has been output.

Program TRACK

TRACK is the root program of the tracks system and, as such, is permanently core-resident when TRACKS is being used. TRACK does no arithmetic processing. Instead, based on the user's responses to various questions, TRACK directs the operation of the overall system by calling various subroutines which then execute in the overlay regions. These subroutines perform the detailed computations required by the analysis indicated by the user.

A major element of TRACK is the various Named Common blocks. By being included in the root segment, these common blocks provide a means of communication between all program elements, regardless of the overlay level at which the elements run. Some of the variables used in these common blocks are initialized in TRACK via Data statements. These include the flash parameters, logical unit numbers, and constants, such as π. It should be noted that initialization of common variables in data statements can only be done in the segment in which the common block first appears.

TRACK proceeds as indicated in the following outline:

CAD, CAM, or DONE?

I. If CAD

   A. Use SETUPS(1) to initialize sums, establish files, and get flash values.

   B. Use PRPROS to calculate section parameters.

C. Use PICTUR to display cross section

D. Find out what to do next.

1. If next section, use SETUPS(2), then loop back to I.B.
2. If new flash, use SETUPS(3), then loop back to I.B.
3. If stress analysis, call FLWSRF and STRESS, then loop back to I.D.
4. If preform, call PFPREP and PREFRM, then loop back to I.D.
5. If done with CAM, call NERGY and RESLTS, then loop back to CAD, CAM, or DONE.

II. If CAM

A. Call SETUPS(3)

B. Call SEQSRF

C. Call PRTSRF

D. Loop back to CAD, CAM, or DONE?

III. If DONE, terminate execution of TRACK and return to monitor.

<u>Subroutine XSECA</u>
<u>(X,Y,RAD,N,AREA,CG,P)</u>

This subroutine calculates the area, perimeter and the X coordinate of the center of gravity of a shape defined as a polygon with a specified radius at each corner or fillet. The mathematical techniques used by this subroutine are described in detail in Appendix II.

Calling sequence parameters:

(1) X and Y: Arrays defining the coordinates of a polygon (input).

(2) RAD: An array of the fillet or corner radii; one value for each coordinate point (input).

(3) N: The number of elements in each of the above arrays (input).

(4) AREA: The cross-section area within the shape defined by X, Y and RAD (output).

(5) CG: The X-coordinate of the center of gravity (output).

(6) P: The perimeter of the section (output).

Function XYINTR (X1,Y1,X2,Y2,X3)

This function returns the second coordinate of a point which lies between the end points (X1, Y1) and (X2, Y2), when the first coordinate of the point, X3, is known. The unknown coordinate is found by linearly interpolating between the end points. The equation is in the form

$$Y3 = X1 + \frac{(X3 - X1)}{(X2 - X1)} * (Y2 - Y1) \ . \qquad \text{(VII-37)}$$

To find the X-coordinate when the Y-coordinate is known, the calling sequence is

$$X3 = XYINTR\ (X1,Y1,X2,Y2,Y3) \ . \qquad \text{(VII-38)}$$

To find the Y-coordinate when the X-coordinate is known, the calling sequence is

$$Y3 = XYINTR\ (Y1,X1,Y2,X2,X3) \ . \qquad \text{(VII-39)}$$

Subroutine YOFFST(Y,Z,RD,YOF,DRCTN,HL)

YOFFST is used by PRTSRF to calculate the Y-axis offsets for the Y-Z planes. The X and Z axis offsets are found in PRTSRF when processing the X-Z data. Y and Z in the calling sequence represent arrays of data for three adjacent points. Referring to Figure VII-33, the angles of the two lines defined by these points are:

$$\theta_B = \tan^{-1}\ ((Z(1) - Z(2))/(Y(1) - Y(2))) \qquad \text{(VII-40)}$$

$$\theta_E = \tan^{-1}\ ((Z(3) - Z(2))/(Y(3) - Y(2))) \quad . \qquad \text{(VII-41)}$$

It should be noted that the values returned for θ_B and θ_E have direction as well as magnitude. The angle of the bisector of the included angle is

$$\theta_A = (\theta_B + \theta_E)/2 \quad . \qquad \text{(VII-42)}$$

Half of the angle subtended by the arc can be found as follows:

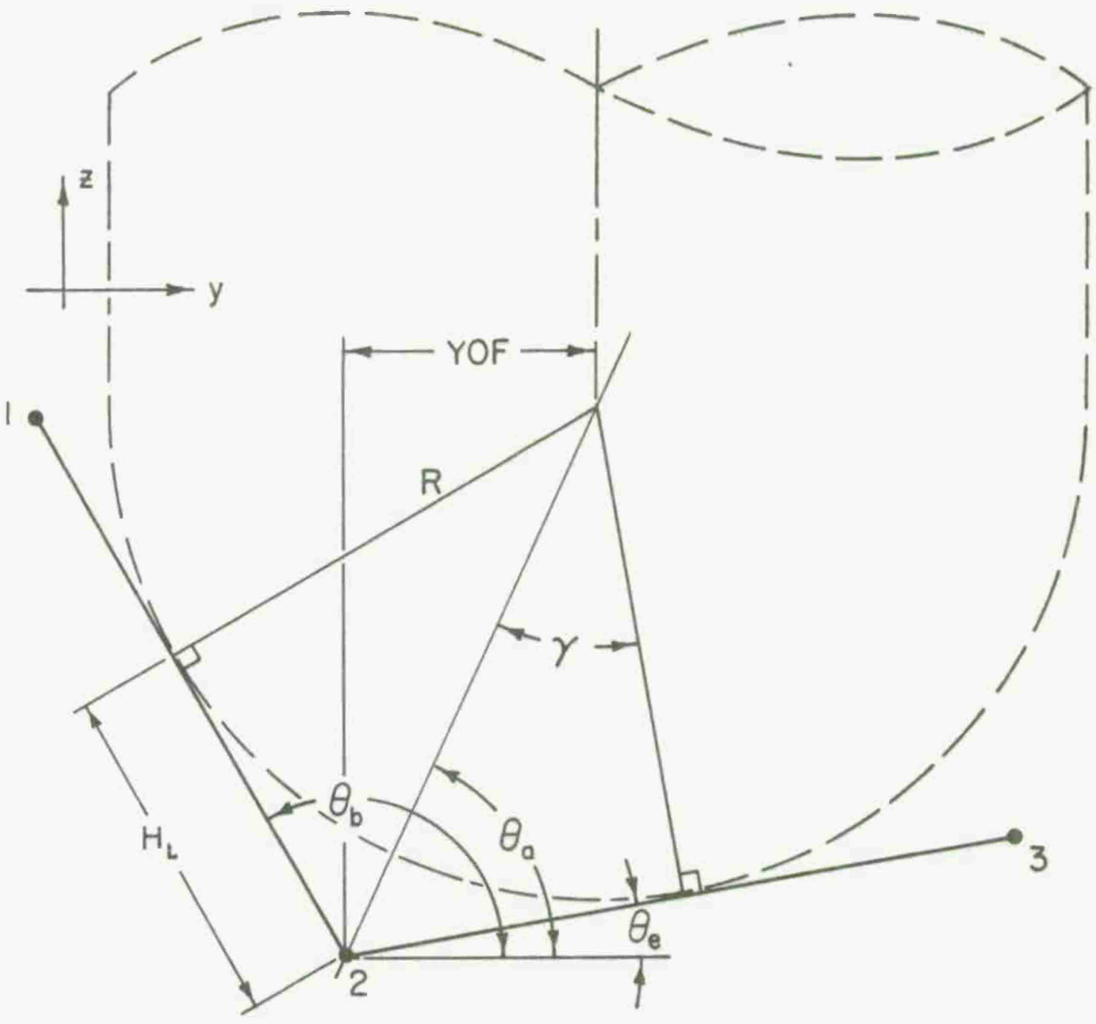


FIGURE VII-33. GEOMETRY TO FIND Y-AXIS TOOL OFFSET

$$\gamma = \pi/2 - (\theta_A - \theta_E)$$

$$= \pi/2 - (\theta_B + \theta_E)/2 + \theta_E \qquad \text{(VII-43)}$$

$$\gamma = (\pi + \theta_E - \theta_B)/2 \quad .$$

The offset value then becomes:

$$YOF = Y(2) + \frac{R}{\cos|\gamma|} \cos \theta_A \cdot D \quad , \qquad \text{(VII-44)}$$

where D = +1 if γ is positive, D = -1 if γ is negative. The value for D is returned in Parameter DRCTN.

Expression V11-44 above is the same general form as used to find the center of an arc fit to the intersection of two lines. However, in case of finding the offset point, the center of the radius (the ball-end mill profiling the surface) always lies above the surface. For fillets, the center and the offset are the same since they both are above the surface. For corners, the center of the arc lies below the surface while the offset is above it. Using D and $|\gamma|$ in Expression VII-44 corrects the center offset so it becomes the tool offset.

Referring to Figure VII-34, Points 1, 2, 3 define a corner to be formed. Calculating the Z offsets gives Surface 1', 2', 3'. If Radius R was fit to this corner, its center would be displayed Y_c to the left of Point 2. However, since the tool offset, YOF, is being found with the tool above the corner, the displacement is made to the right of Point 2'. Thus, Point 2' is shifted to 2". Figure VII-35 shows the surface gouging which would occur without Y-axis compensation for the tool radius.

The parameters in the calling sequence are as follows:

Y,Z:	Arrays defining three consecutive points on a polygon.
RD:	The radius of the ball-end mill to be used to machine the surface.
YOF:	The Y-axis offset position (absolute coordinate).
DRCTN:	+1, if a fillet point; -1, if a corner point.
ML:	Distance from the point on the polygon to tangent of the radius with the polygon.

Named Common block SYSPR is also used to transfer data.

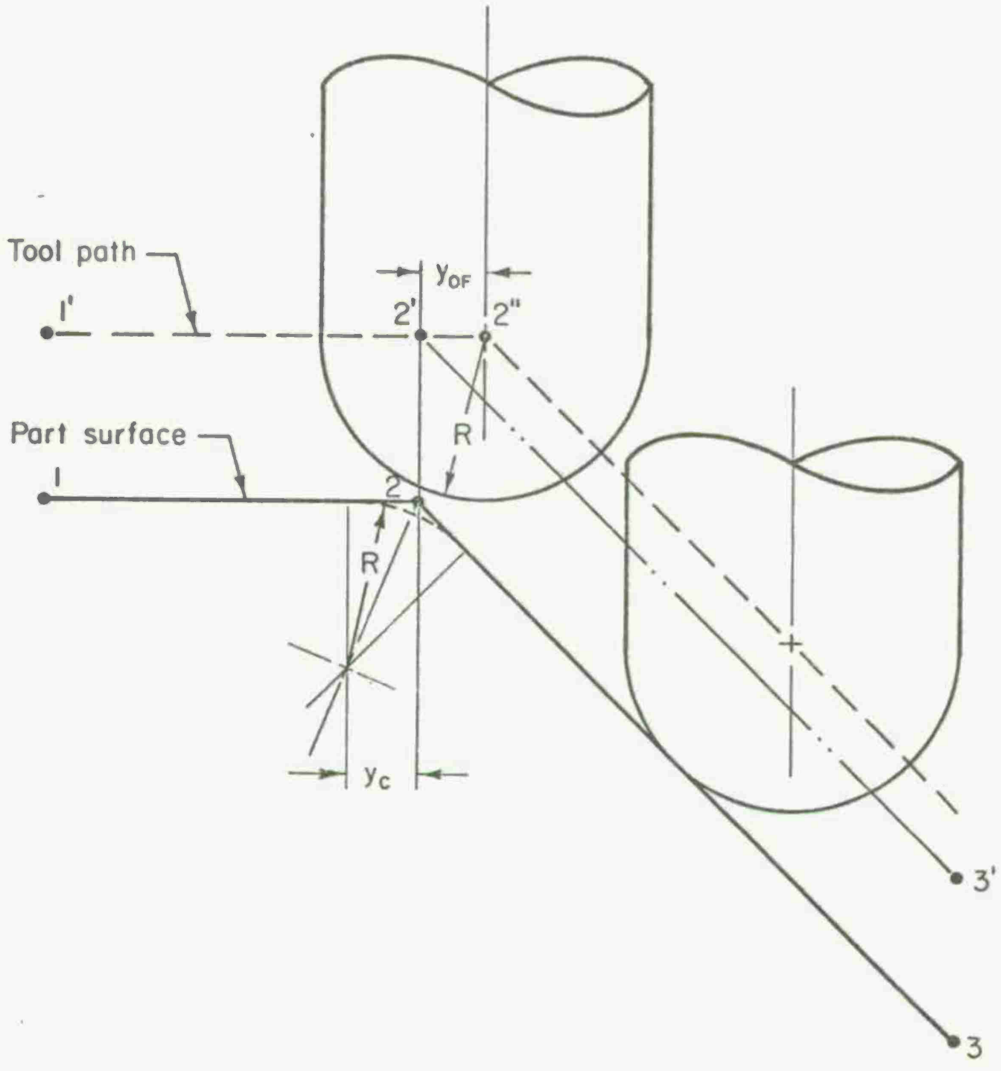


FIGURE VII-34. RELATIONSHIP OF THE CENTER OF A CORNER AND Y-A AXIS TOOL OFFSET

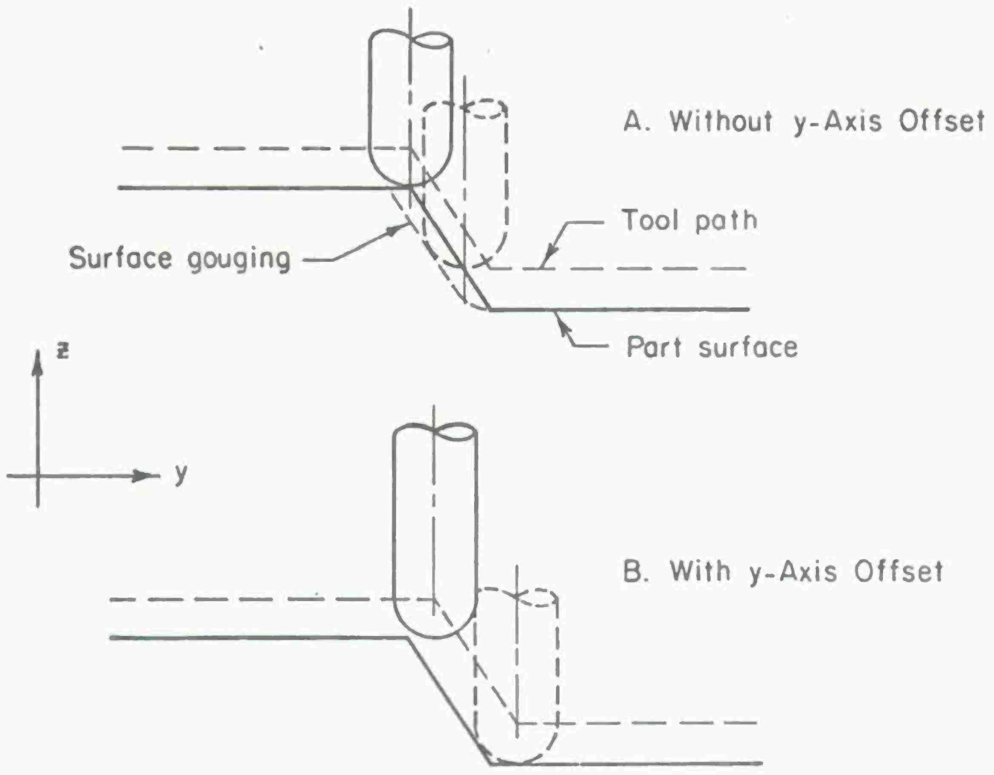


FIGURE VII-35. TOOL PATHS WITH AND WITHOUT Y-AXIS TOOL OFFSET COMPENSATION

APPENDIX VIII

DESCRIPTION OF "COMMON" BLOCKS

APPENDIX VIII

DESCRIPTION OF "COMMON" BLOCKS

The following describes the variables associated with each "Common" block. "Common" blocks are used by FORTRAN as a means of communication between program elements, such as between subroutines or overlay segments.

Blank Common

1. XPLOTT, ZPLOTT - 250 element arrays for the coordinates defining the interpolated upper (top) die surface.
2. NPLOTT - The number of elements used to define the upper part surface.
3. XPLOTB, ZPLOTB - 250 element arrays for the coordinates defining the interpolated lower (bottom) part surface.
4. NPLOTB - The number of elements used to define the lower part surface.

Labeled Common /DISPLA/

1. ZMAX - The value of the coordinate point having the greatest vertical (z - axis) value.
2. ZMIN - The value of the coordinate point with the smallest vertical (z - axis) value.
3. XMAX - The value of the coordinate point with the greatest horizontal (x - axis) value.
4. XMIN - The value of the coordinate point with the smallest horizontal (x - axis) value.
5. NLINES - The number of lines reserved for the text scroller area when text is displayed along with graphic images.
6. AL - The vertical position, in raster units, where the text scroller will be positioned.
7. SCALEF - The scale factor used to convert from object (user) space dimensions to subject space dimensions (CRT raster units).
8. AZ - Unused.
9. NTAG - The tag value of the next item to be displayed.
10. DISBFR - A 3000 word array used as the display buffer.

11. GRPHCS - A logical (byte) variable used to indicate that the graphics subroutines have been initialized. Initialized, FALSE.

Labeled Common /DEFEL/

A. When used for preform design:
   (1) XPF(50), ZPF(50), RPF(50) - 3, 50-element arrays for the coordinates of the preform polygon.
   (2) NPF - the number of coordinate points defining the preform polygon.
   (3) K(10) - a 10-element array used to return the indices of the points indicated with the L.P. in subroutine GETHIT.
   (4) XPFPLT(250), ZPFPLT(250) - 250-element arrays used to contain the coordinates of the interpolated preform.

B. When used for stress analysis:
   (1) XDEFEL - a 100-element array defining the X coordinate of trapaxoidal deformation elements.
   (2) ZDEFLT, ZDEFLB - 100-element arrays defining the upper and lower Z coordinates of the deformation elements.
   (3) FDEFLT, FDEFLB - 100-element arrays defining the upper and lower flow factor operating on the surface of each deformation element.
   (4) NDEFEL - the number of deformation elements found for the section.
   (5) SYPLOT - 100-element arrays for the stress value at each deformation element.

Labeled Common /FINISH/

1. XPST, ZPST, RPST - 30-element arrays for the coordinates and corner or fillet radii, defining the upper surface polygon including flash coordinates. Result of subroutine DIESRF.
2. NPST - The number of elements used to define the upper die surface polygon.
3. XPSB, ZPSB, RPSB - 30-element arrays for the coordinates, and corner or fillet radii, defining the lower die surface polygon including

the flash coordinates. Result of subroutine DIESRF.

4. NPSB - The number of elements used to define the lower die surface polygon.

<u>Labeled Common/SECTN/</u>

1. LUNFIN - The logical unit number associated with the device used to store the coordinate file describing the finish die. Assigned a value of 3 via a Data statement.
2. LUNBLK - The logical unit number associated with the device used to store the coordinate file, defining the preform part. Assigned a value of 4 via a Data statement.
3. LUNNC - The logical unit number associated with the device used to store the NC tape image file for the preform dies.
4. SAVFIN - A logical (byte) variable used to indicate that the finish die coordinate file has been saved on unit LUNFIN.
5. SAVBLK - A logical (byte) variable used to indicate that the preform part coordinate file has been saved on unit LUNBLK.
6. SAVENC - A logical (byte) variable used to indicate that the interpolated upper and lower die profiles have been saved on the NC preparation file.
7. NEXT - A logical (byte) variable used to indicate that the next section in the input data file is to be processed.
8. SUMARE - The total plan area for all sections processed and saved as output files.
9. SUMPVL - The total volume for all sections processed, excluding the flash volume.
10. SUMTVL - The total volume for all sections processed, including the flash volume.
11. SUMLD - The total load on all sections processed.
12. SUMXLD - The X-coordinate of the center of load for all sections processed.
13. SUMYLD - The Y-coordinate of the center of load for all sections processed.
14. BLTARE - The cross-sectional area for the largest section processed.
15. ENERGY - The energy required to forge the model part.
16. GUTMAX - The maximum X-axis dimension of any section to be processed, plus an allowance for flash and clearance.

17. GUTMIN - The minimum X-axis dimension of any section to be processed, plus an allowance for flash and clearance.
18. XR - A 50-element array for the X-coordinates of the part section polygon, rotated into the X-Z plane.
19. X - A 51-element array for the X-coordinates of the part section polygon. Read from cards.
20. YR - The position of the section polygon along the Y-axis, after the polygon is rotated into the X-Z plane.
21. Y - A 51-element array for the Y-coordinates of the section polygon. Read from cards.
22. ZR - A 51-element array for the Z-coordinates of the section polygon. Read from cards.
23. RR - A 51-element array for the corner or fillet radii of the section polygon. Read from cards.
24. NP - The number of coordinate points defining the section polygon.
25. WTDWID - The sum of the width times volume for each section analyzed. Used to determine cavity width of energy model.
26. SUMDPH - The sum of the length of flash of each section.
27. PLN - An unused 2-element array.
28. ISCTNO - An identifying number for the section. Read from the section header card.
29. ITYPE - An identifier as to the type of section. Read from the section header card. 1 = plane strain, 2 = axisymmetric strain with single flash, 3 = axisymmetric strain with double flash.
30. DEPANG - The depth (thickness) of the section in plane strain; the wedge angle of the section in axisymmetric strain. Read from section header card.
31. AXISYM - The X-coordinate of the axis of symmetry when the section has been rotated to the X-Z plane.
32. ANGL - The wedge angle in radians, if an axisymmetric section.
33. IERROR - Indicates various error conditions. If equal to 1, no error.
34. XAREA - The cross-sectional area of the section (does not include flash).
35. PLAREA - The plan area of the section, including the flash.
36. PERIM - The perimeter of the section, without flash.
37. PRTVOL - The volume of the section, without flash.
38. FLSVOL - The flash volume of the section.

39. TOTVOL - The combined part and flash volumes of the section.
40. XCNTRD - The X-coordinate of the center of mass of the section, in the original coordinate system.
41. YCNTRD - The Y-coordinate of the center of mass of the section, in the original coordinate system.
42. CNSTDS - The Y-coordinate of the section plane after the plane is rotated parallel to the X-Z plane.
43. ANGROT - The angle of the original section plane relative to the X-Z plane.
44. IRHPL - The index to the right-hand parting line coordinates for the input polygon.
45. FLSTHK - The flash thickness. Default value is 0.125 inches.
46. FLSWID - The flash width. Default value is 0.250 inches.
47. FLSRAD - The flash radius. Default value is 0.125 inches.
48. FLSFTR - A dimensionless factor used to account for the material squeezed out from between the dies, beyond the limits of the flash width. Default value is 4.0.
49. XNUTRL - The X-coordinate of the neutral surface.
50. STRSMX - The maximum stress in a section.
51. SUMF - The total vertical load on a section.
52. XCOFL - The X-coordinate of the center of load on a section.
53. PFVOL - The volume of the preform of the section.
54. DIF - The difference in volume between the preform and the finished part. DIF = PFVOL - TOTVOL.
55. PFRHPL - The index to the coordinate defining the right-hand parting line point of the preform.
56. FS - Material flow stress.
57. EXCOEF - Thermal expansion coefficient (subroutine SAVPLY)
58. FRCTON - Coefficient of friction.
59. STMXMX = The maximum stress found for any section analyzed.

Labeled Common /STRSPT/

1. XTOP, ZTOP - 11-element arrays for the coordinates of shear boundaries on the upper die surface.
2. NTOP - The number of shear boundaries on the upper die surface, plus one. Default is 1.
3. XBOT, ZBOT - 11-element arrays for the coordinates of shear boundaries on the lower die surface.
4. NBOT - The number of shear boundaries on the lower die surface, plus one. Default is 1.

Labeled Common/SHRBND/

1. XINTT, ZINTT - 11-element arrays for the coordinates of points on shear surfaces on the upper die surface.
2. FLAGT - A logical variable that, when true, indicates that the upper die is in axisymmetric, single flash strain with a shear boundary on the axis of symmetry. Default is false.
3. XINTB, ZINTB - 11-element arrays for the coordinates of points on shear surfaces on the lower die surface.
4. FLAGB - A logical variable that, when true, indicates that the lower die is in axisymmetric, single flash strain with a shear boundary on the axis of symmetry. Default is false.
5. FRICTN - A 2-element array for the flow factors in friction and shear. Default values are 0.25 and 0.577, respectively.
6. XSHRT, ZSHRT - 50-element arrays for the coordinates defining the upper flow surface.
7. NSHRT - The number of elements used to define the upper flow surface.
8. XSHRB, ZSHRB, FSHRB - 50 element arrays for the coordinates defining the lower flow surface.
9. NSHRB - The number of elements used to define the lower flow surface.

Labeled Common /SYSPR/

1. LUNKB - The keyboard logical unit number. Given a value of 5 via DATA statement.
2. LUNCRT - The CRT logical unit number. Given a value of 5 via DATA statement.
3. LUNPR - The line printer logical unit number. Given a value of 1 via DATA statement.
4. LUNCR - The card reader logical unit number. Given a value of 2 via DATA statement.
5. PI - The numerical value of pi (π).
6. IPRINT - Used to control the printing of intermediate results. Default = 1. (No intermediate results will be printed).
7. ZERO - The numerical value of zero (0.0).

APPENDIX IX

UTILITY PROGRAMS

APPENDIX IX

UTILITY PROGRAMS

A number of stand-alone programs were developed to assist in preparing input for, or evaluating the output from the track shoe CAD/CAM system. These programs are briefly described below.

Program NCDATA

NCDATA displays an NC data file on the CRT. The data displayed is the same as is used by the BCL NC milling machine to produce the preform model surface. By displaying the cutter paths on the CRT, gross errors may be checked for. It should be noted that NCDATA displays the center-line paths of the ball-end mill specified for the part, and does not display the true part surface. However, when the cutter size is small (.5-inch diameter or less), the center-line paths displayed are a very close approximation to the true surface.

A feature of NCDATA is its ability to show the cutter paths in true isometric projection from any position relative to the first coordinate. This allows the viewer to "walk around" the surface and view it from any position. The view may also be scaled either up or down to permit the displayed image to appear as large as possible. The viewing position and scale must be specified by the user before the image is displayed. After the image is generated, the user may modify these parameters in order to view the surface from another position or at another scale.

Once the image is displayed, the user may translate the image both horizontally and vertically on the CRT. This is done using the light pen. Often, it is difficult to judge beforehand how an image should be positioned so that the entire image appears on the screen. Translation via the light pen allows the image to be centered on the screen after it is generated, in case the starting coordinates place part of the image off of the screen.

The following summarizes the inputs required from the user. The numerical responses (underlined values) are the values entered to generate the surface shown in Figure IX-1.

(1) File Name: This is the name of the NC output file generated by the CAM phase of the track-shoe system (subroutine PRTSRF).

(2) Absolute or Incremental Data Format: All data generated by the track-shoe system is incremental.

(3) Scale Factor: The amount by which the data is to be enlarged or reduced in order to fill the screen without going off the edges (0.7).

(4) Viewing Position: The coordinate position of the viewer along the three axis, relative to the first point of the display (-1, -1, 1).

(5) Starting Position of the Display on the CRT in Screen (Raster) Units: This locates the first point of the image and all other points are relative to it (500, 700).

When the image is fully displayed, and after the light pen translation operation is completed, the viewer is given the opportunity to copy the displayed image onto the X-Y recorder. This provides him a hard-copy of the image he is viewing. After the image is copied, or if this step is skipped, the user is given the opportunity to generate a new display from the same file, but at a different scale or viewing position. In order to view a different file, it is necessary to terminate the current execution of NCDATA, and to re-RUN it.

Due to memory limitations on the size of the display file which can be handled, NCDATA cannot display files which are larger than 70 disk blocks. If the display of a file larger than this is attempted, the display will start, but will terminate when the limit is reached. For files larger than 70 blocks, NCPLOT should be used.

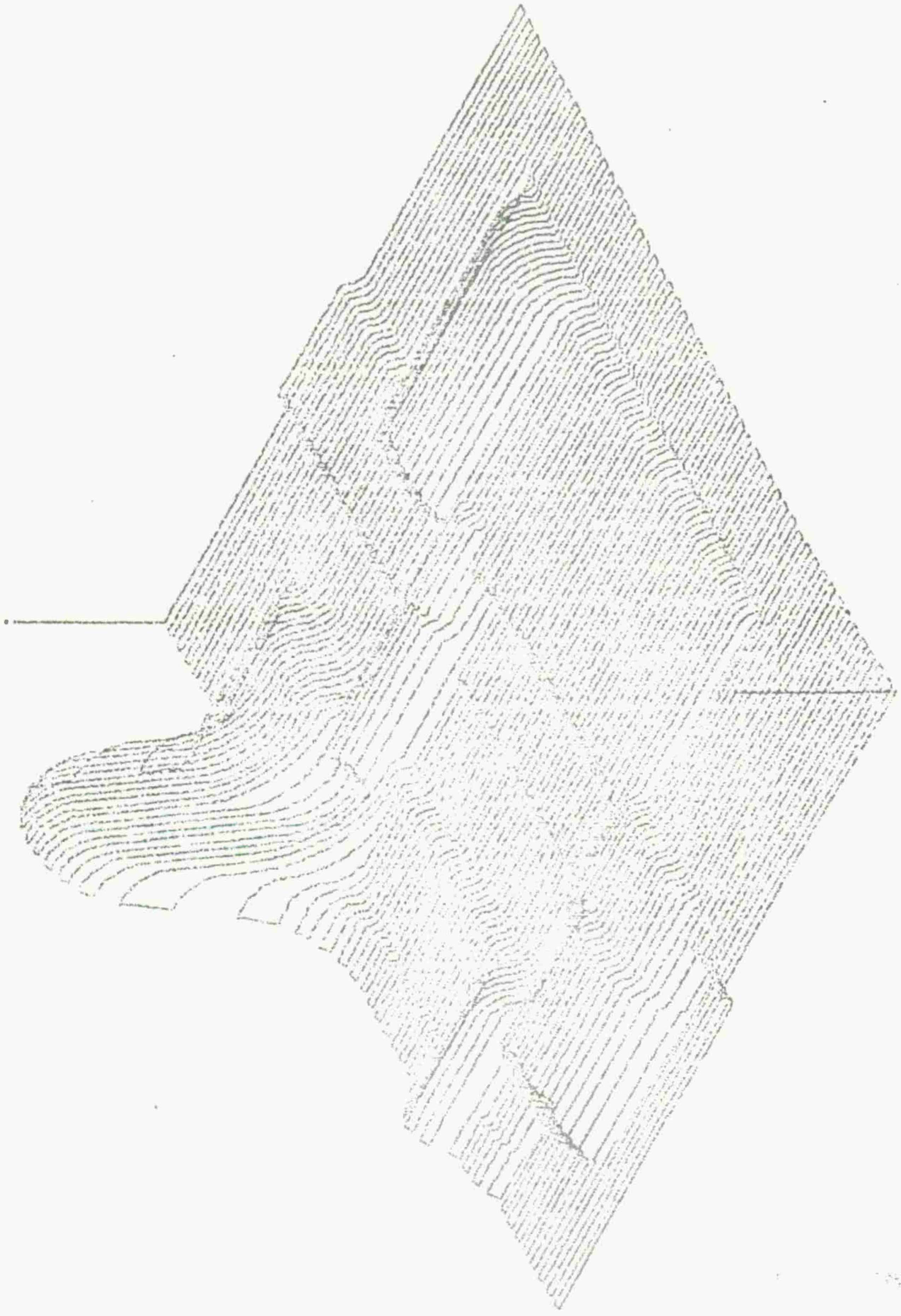


FIGURE IX-1. PLOT OF CUTTER PATHS GENERATED BY PROGRAM NCDATA

Program NCPLOT

NCPLOT was derived from NCDATA and has the features of NCDATA except that it does not generate a display file on the CRT. NCPLOT outputs the specified file directly to the X-Y recorder. NCPLOT requires the same information from the user as is needed by NCDATA. A natural limitation of NCPLOT, being hard copy only, is that light pen translation of the image is not possible. If translation is necessary, the user must generate another plot and specify different starting coordinates. The primary use of NCPLOT is to plot data files which are too large for NCDATA. NCPLOT can handle a file of, essentially, unlimited size.

Program ENTRDT

ENTRDT allows the user to create a data file in the format required by the track-shoe system. It asks the user to enter the various values needed for a data file. The questions asked are as follows:

(1) Name of the file to be created. This must be entered as required by standard DEC procedures for file names (i.e., T130SH.DAT).

(2) Eighty(80) alpha-numeric characters for the file header. This could be the part name and number, or other text information which describes the part.

(3) The section number and type identifier as integer values. Four digits are allowed for the section number. The section type identifier may be 1, 2, or 3 for plane strain, axisymmetric single flash and axisymmetric double flash sections, respectively. If a negative section number is entered, ENTRDT closes the data file and terminates.

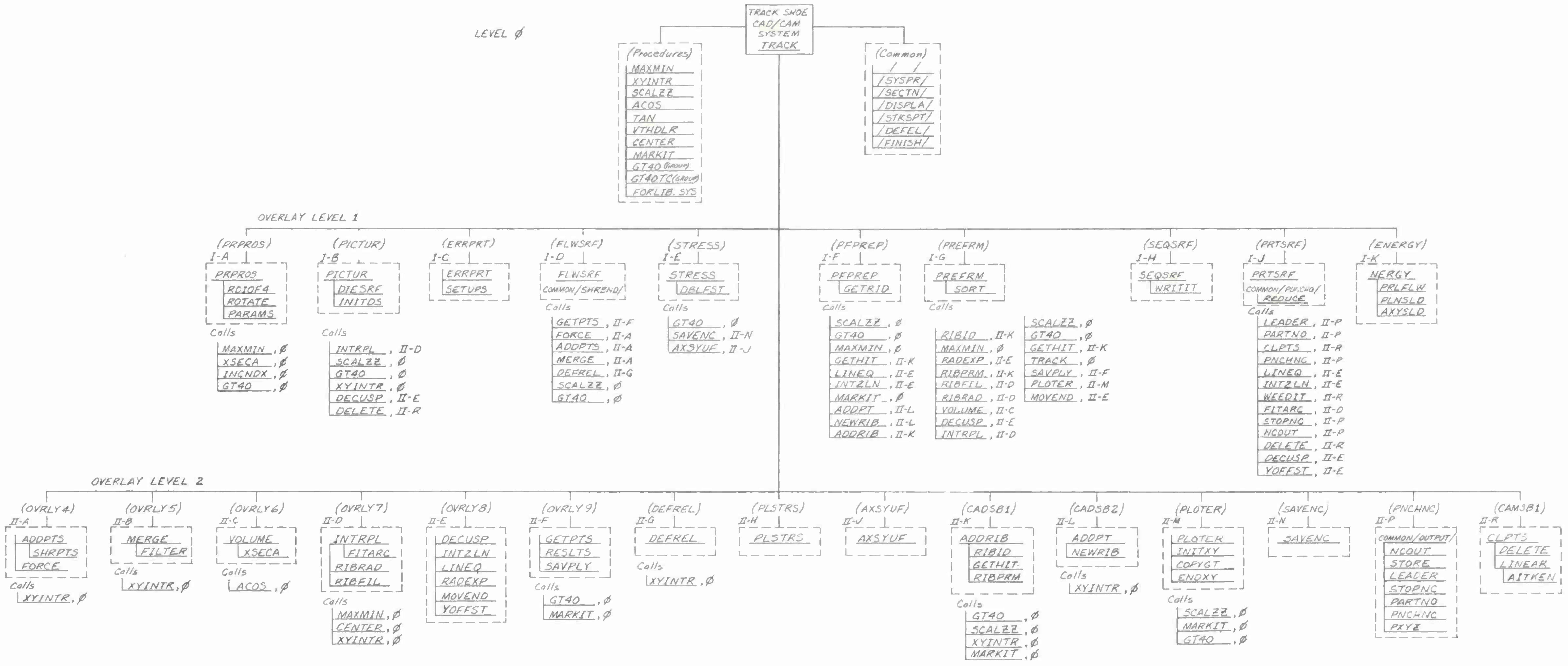
(4) The depth or wedge angle of the section. Integer or decimal values with a maximum of three decimal accuracy.

(5) The X, Y, Z coordinates and radius value for each point on the polygon. These may either be integer or up to three-place decimal values. The four values are entered on the same line with comma separators. The program continues to

accept the coordinate data, four values at a time, until a value greater than 9000 is entered for x. When this is found, a section terminator is created in the file and the program loops back and asks for the next section number (Item 3 above).

APPENDIX X

PROGRAM MODULE ORGANIZATION



TRACK SHOE CAD/CAM SYSTEM TRACK
LEVEL Ø
(Procedures)
MAXMIN
XYINTR
SCALZZ
ACOS
TAN
VTHDLR
CENTER
MARKIT
GT40 (GROUP)
GT40TC (GROUP)
FORLIB.SYS
(Common)
/ /
/SYSPR/
/SECTN/
/DISPLA/
/STRSPT/
/DEFEL/
/FINISH/
OVERLAY LEVEL 1
(PRPROS) I-A
PRPROS
RDIOF4
ROTATE
PARAMS
Calls
MAXMIN, Ø
XSECA, Ø
INCNDX, Ø
GT40, Ø
(PICTUR) I-B
PICTUR
DIESRF
INITDS
Calls
INTRPL, II-D
SCALZZ, Ø
GT40, Ø
XYINTR, Ø
DECUSP, II-E
DELETE, II-R
(ERRPRT) I-C
ERRPRT
SETUPS
(FLWSRF) I-D
FLWSRF
COMMON/SHREND/
Calls
GETPTS, II-F
FORCE, II-A
ADDPTS, II-A
MERGE, II-A
DEFREL, II-G
SCALZZ, Ø
GT40, Ø
(STRESS) I-E
STRESS
DBLFST
Calls
GT40, Ø
SAVENC, II-N
AXSYUF, II-J
(PFPREP) I-F
PFPREP
GETRID
Calls
SCALZZ, Ø
GT40, Ø
MAXMIN, Ø
GETHIT, II-K
LINEQ, II-E
INT2LN, II-E
MARKIT, Ø
ADDPT, II-L
NEWRIB, II-L
ADDRIB, II-K
(PREFRM) I-G
PREFRM
SORT
Calls
RIBID, II-K
MAXMIN, Ø
RADEXP, II-E
RIBPRM, II-K
RIBFIL, II-D
RIBRAD, II-D
VOLUME, II-C
DECUSP, II-E
INTRPL, II-D
SCALZZ, Ø
GT40, Ø
GETHIT, II-K
TRACK, Ø
SAVPLY, II-F
PLOTER, II-M
MOVEND, II-E
(SEQSRF) I-H
SEQSRF
WRITIT
(PRTSRF) I-J
PRTSRF
COMMON/PUPSHO/
REDUCE
Calls
LEADER, II-P
PARTNO, II-P
CLPTS, II-R
PNCHNC, II-P
LINEQ, II-E
INT2LN, II-E
WEEDIT, II-R
FITARC, II-D
STOPNC, II-P
NCOUT, II-P
DELETE, II-R
DECUSP, II-E
YOFFST, II-E
(ENERGY) I-K
NERGY
PRLFLW
PLNSLD
AXYSLD
OVERLAY LEVEL 2
(OVRLY4) II-A
ADDPTS
SHRPTS
FORCE
Calls
XYINTR, Ø
(OVRLY5) II-B
MERGE
FILTER
Calls
XYINTR, Ø
(OVRLY6) II-C
VOLUME
XSECA
Calls
ACOS, Ø
(OVRLY7) II-D
INTRPL
FITARC
RIBRAD
RIBFIL
Calls
MAXMIN, Ø
CENTER, Ø
XYINTR, Ø
(OVRLY8) II-E
DECUSP
INT2LN
LINEQ
RADEXP
MOVEND
YOFFST
(OVRLY9) II-F
GETPTS
RESLTS
SAVPLY
Calls
GT40, Ø
MARKIT, Ø
(DEFREL) II-G
DEFREL
Calls
XYINTR, Ø
(PLSTRS) II-H
PLSTRS
(AXSYUF) II-J
AXSYUF
(CADSB1) II-K
ADDRIB
RIBID
GETHIT
RIBPRM
Calls
GT40, Ø
SCALZZ, Ø
XYINTR, Ø
MARKIT, Ø
(CADSB2) II-L
ADDPT
NEWRIB
Calls
XYINTR, Ø
(PLOTER) II-M
PLOTER
INITXY
COPYGT
ENDXY
Calls
SCALZZ, Ø
MARKIT, Ø
GT40, Ø
(SAVENC) II-N
SAVENC
(PNCHNC) II-P
COMMON/OUTPUT/
NCOUT
STORE
LEADER
STOPNC
PARTNO
PNCHNC
PXYZ
(CAM3B1) II-R
CLPTS
DELETE
LINEAR
AITKEN

Army Materials and Mechanics Research Center
Watertown, Massachusetts 02172
COMPUTER-AIDED DESIGN (CAD) AND COMPUTER-AIDED MANUFACTURING (CAM) FOR CLOSED-DIE FORGING OF TRACK SHOES AND LINKS
Carl F. Billhardt, Nuri Akgerman, and T. Altan, Battelle's Columbus Laboratories, Columbus, Ohio 43201
Technical Report, August 1976, 224 pp - illus -Appendixes, Contract DAAG46-75-C-0041
Final Report, November 11, 1975 to May 10, 1976

AD ______________________

UNCLASSIFIED
UNLIMITED DISTRIBUTION

Key Words

Computer-Aided Design
CAD/CAM
Forging
Steel
Numerical Control
NC
Computer Graphics
Die Design

The overall objectives of this program were (a) to reduce manufacturing costs and improve productivity in forging track shoes and links, and (b) to reduce the lead times, which are traditionally long in supplying forged parts. With successful implementation of the results of this program, potential cost reductions (including material savings, die cost reduction, and reduction of machining) could be substantial since track shoes and links are produced in very large quantities. In addition, the potential benefits of the CAD/CAM techniques, developed in this program, include the improvement of dimensional consistency, and reduction of lead times for new and modified track-shoe designs.

Army Materials and Mechanics Research Center
Watertown, Massachusetts 02172
COMPUTER-AIDED DESIGN (CAD) AND COMPUTER-AIDED MANUFACTURING (CAM) FOR CLOSED-DIE FORGING OF TRACK SHOES AND LINKS
Carl F. Billhardt, Nuri Akgerman, and T. Altan, Battelle's Columbus Laboratories, Columbus, Ohio 43201
Technical Report, August 1976, 224 pp - illus -Appendixes, Contract DAAG46-75-C-0041
Final Report, November 11, 1975 to May 10, 1976

AD ______________________

UNCLASSIFIED
UNLIMITED DISTRIBUTION

Key Words

Computer-Aided Design
CAD/CAM
Forging
Steel
Numerical Control
NC
Computer Graphics
Die Design

The overall objectives of this program were (a) to reduce manufacturing costs and improve productivity in forging track shoes and links, and (b) to reduce the lead times, which are traditionally long in supplying forged parts. With successful implementation of the results of this program, potential cost reductions (including material savings, die cost reduction, and reduction of machining) could be substantial since track shoes and links are produced in very large quantities. In addition, the potential benefits of the CAD/CAM techniques, developed in this program, include the improvement of dimensional consistency, and reduction of lead times for new and modified track-shoe designs.

DISTRIBUTION LIST

No. of Copies	To
1	Office of the Director, Defense Research and Engineering, The Pentagon, Washington, D. C. 20301
12	Commander, Defense Documentation Center, Cameron Station, Building 5, 5010 Duke Street, Alexandria, Virginia 22314
1	Advanced Research Projects Agency, The Pentagon, Washington, D. C. 20315
	Metals and Ceramics Information Center, Battelle Memorial Institute, 505 King Avenue, Columbus, Ohio 43201
2	ATTN: Mr. Daniel Maykuth
	Chief of Research and Development, Department of the Army, Washington, D. C. 20310
2	ATTN: Physical and Engineering Sciences Division
1	Dr. Bernard R. Stein
	Commander, U. S. Army Materiel Development and Readiness Command, 5001 Eisenhower Avenue, Alexandria, Virginia 22333
1	ATTN: DRCDE-DE, Development Division
1	DRCDE-RS, Research Division
1	DRCDE-RS, Scientific Deputy
1	DRCDE-TC
	Commander, U. S. Army Aviation Systems Command, P. O. Box 209, Main Office, St. Louis, Missouri 63166
1	ATTN: SRSAV-LEP, Mr. J. M. Thorp
1	SRSAV-ER, Dr. I. Peterson
	Commander, U. S. Army Missile Command, Redstone Arsenal, Alabama 35809
1	ATTN: DRSMI-IE, Mr. J. E. Kirshtein
1	DRSMI-R, Mr. John L. McDaniel
1	DRSMI-RBLD, Redstone Scientific Information Center
1	Chief Scientist, Dr. W. W. Carter
1	Directorate of R&D
1	Dr. B. Steverding
	Commander, U. S. Army Mobility Equipment Command, 4300 Goodfellow Boulevard, St. Louis, Missouri 63120
1	ATTN: AMSME-PLC, Mr. J. Murphy
	Commander, U. S. Army Armament Command, Rock Island, Illinois 61201
1	ATTN: AMSAR-SC, Dr. C. M. Hudson
1	AMSAR-PPW-PB, Mr. Francis X. Walter
2	Technical Library

No. of Copies	To
	Commander, U. S. Army Tank-Automotive Development Center, Warren, Michigan 48090
1	ATTN: DRDTA-PPS, Mr. David Siegel
1	Mr. J. P. Jones
	Commander, Aberdeen Proving Ground, Maryland 21005
3	ATTN: Technical Library, Building 313
	Commander, U. S. Army Foreign Science and Technology Center, 220 7th Street, N. E., Charlottesville, Virginia 22901
1	ATTN: DRXST-SD3
	Commander, Frankford Arsenal, Philadelphia, Pennsylvania 19137
1	ATTN: Pitman-Dunn Institute of Research
1	SARFA-L300, Mr. J. Corrie
	Commander, Picatinny Arsenal, Dover, New Jersey 07801
1	ATTN: Feltman Research Laboratories
	Commander, Rock Island Arsenal, Rock Island, Illinois 61201
1	ATTN: SARRI-RDL
	Director, Eustis Directorate, U. S. Army Air Mobility Research and Development Laboratory, Fort Eustis, Virginia 23604
1	ATTN: Mr. J. Robinson, SAVDL-EU-SS
	Commander, U. S. Army Ballistic Research Laboratories, Aberdeen Proving Ground, Maryland 21005
1	ATTN: Dr. D. Eichelberger
	Director, U. S. Army Materiel Systems Analysis Activity, Aberdeen Proving Ground, Maryland 21005
1	ATTN: DRXRD-D
	Commander, U. S. Army Electronics Command, 225 South 18th Street, Philadelphia, Pennsylvania 19103
1	ATTN: DRSEL-PP/P/IED-2, Mr. Wesley Karg
	Commander, U. S. Army Mobility Equipment Research and Development Command, Fort Belvoir, Virginia 22060
2	ATTN: Technical Documents Center, Building 315
	Commander, U. S. Army Production Equipment Agency, Manufacturing Technology Branch, Rock Island Arsenal, Illinois 61202
1	ATTN: AMXPE, Mr. Ralph Siegel

No. of Copies	To
	Commander, U. S. Army Research and Engineering Directorate, Warren, Michigan 48090
2	ATTN: SMOTA-RCM.1, Mr. Edward Moritz
1	SMOTA-RCM.1, Mr. Donald Phelps
	Commander, Watervliet Arsenal, Watervliet, New York 12189
1	ATTN: SARWV-R
1	Dr. Robert Weigle
1	Chief, Bureau of Naval Weapons, Department of the Navy, Room 2225, Munitions Building, Washington, D. C.
	Chief, Bureau of Ships, Department of the Navy, Washington, D. C. 20315
1	ATTN: Code 341
	Chief of Naval Research, Arlington, Virginia 22217
1	ATTN: Code 471
	Naval Research Laboratory, Washington, D. C. 20375
2	ATTN: Dr. G. R. Yoder - Code 6382
	Headquarters, USAF/RDPI, The Pentagon, Washington, D. C. 20330
1	ATTN: Major Donald Sponberg
	Headquarters, Aeronautical Systems Division, 4950 TEST W/TZHM (DH 2-5 Mgr), Wright-Patterson Air Force Base, Ohio 45433
1	ATTN: AFML/MATB/Mr. George Glenn
2	AFML/MXE/E. Morrissey
1	AFML/LLP/D. M. Forney, Jr.
1	AFML/LC
1	AFML/MBC/Mr. Stanley Schulman
	National Aeronautics and Space Administration, Washington, D. C. 20546
1	ATTN: AFSS-AD, Office of Scientific and Technical Information
1	Mr. B. G. Achhammer
1	Mr. G. C. Deutsch, Chief, Materials Research Program, Code RR-1
	National Aeronautics and Space Administration, Lewis Research Center, 21000 Brookpark Road, Cleveland, Ohio 44135
1	ATTN: Mr. G. Mervin Ault, Assistant Chief, M&S Division
	National Aeronautics and Space Administration, Marshall Space Flight Center, Huntsville, Alabama 35812
1	ATTN: S&E-ME-MM, Mr. W. A. Wilson, Building 4720
1	R-P&VE-M, R. J. Schwinghamer
	Albany Metallurgy Research Center, Albany, Oregon 97321
1	ATTN: Mr. A. H. Roberson, Research Director

No. of Copies	To
	Defense Materials Service, General Services Administration, Washington, D. C. 20405
1	ATTN: Mr. Clarence A. Fredell, Director, Technical R&D Staff
	General Dynamics, Convair Aerospace Division, P. O. Box 748, Fort Worth, Texas 76101
1	ATTN: Mfg. Engineering Technical Library
	Director, Army Materials and Mechanics Research Center, Watertown, Massachusetts 02172
2	ATTN: DRXMR-PL
1	DRXMR-PR
1	DRXMR-CT
1	DRXMR-AP
1	DRXMR-XC
1	DRXMR-M
12	DRXMR-ER